Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри»

Виконав: ст. 2-го курсу, групи КІТС-22 мз
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем
(шифр і назва напряму підготовки, спеціальності)

_____ Ратушняк С.С. _____
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. МБІС
_____ Грицак А.В. _____
(прізвище та ініціали)
«____» _____ 2024 р.

Опонент: к.т.н., доцент, каф. ОТ
_____ Крупельницький Л.В. _____
(прізвище та ініціали)
«____» _____ 2024 р.

Допущено до захисту
Голова секції УБ кафедри МБІС

_____ Юрій ЯРЕМЧУК
“_____” _____ 2024 р.

Вінниця ВНТУ – 2024 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

_____ **Юрій ЯРЕМЧУК**
“_____” _____2024 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу студенту
Ратушняк Сергій Сергійович
(прізвище, ім`я, по-батькові)

1. Тема роботи:

«Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри»
Керівник роботи_: Грицак А.В., к.т.н., доцент каф. МБІС
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “ 11” березня 2024 року № 81
2. Строк подання студентом роботи_за тиждень до захисту.__
3. Вихідні дані до роботи:
Стандарти, електронні джерела, підручники та наукові статті по темі, як стосуються теми магістерської кваліфікаційної роботи.
4. Зміст текстової частини:
У першому розділі розглянуто теоретичні засади підвищення захищеності системи моніторингу та управління IoT-пристроями на основі моделі нульової довіри (Zero Trust) із використанням алгоритму динамічного встановлення довіри. У другому розділі проведено проектування захищеної системи моніторингу та управління IoT-пристроями. У третьому розділі здійснено програмну реалізацію розробленої системи. Обґрунтовано вибір мови реалізації та середовища розробки, проведено реалізацію системи моніторингу та управління IoT-пристроями. Четвертий розділ містить економічне обґрунтування розробки програмного забезпечення, включаючи оцінювання комерційного потенціалу.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)

У дугому розділі магістерської кваліфікаційної роботи наведено 5 рисунків, у третьому розділі – 12 рисунків

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина			
I	Грицак А.В., к.т.н., доцент каф. МБІС		
II	Грицак А.В., к.т.н., доцент каф. МБІС		
III	Грицак А.В., к.т.н., доцент каф. МБІС		
Економічна частина			
IV	Причепа І.В., к.е.н., доц. каф. ЕПВМ		

7.Дата видачі завдання _____2024 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1.	Визначення напрямку магістерської роботи, формулювання теми	1.03.2024	15.03.2024	
2.	Аналіз предметної області обраної теми	15.03.2024	1.04.2024	
3.	Розробка роботи	1.04.2024	10.04.2024	
4.	Написання магістерської роботи на основі розробленої теми	10.04.2024	20.04.2024	
5.	Передзахист магістерської кваліфікаційної роботи	20.04.2024	10.05.2024	
6.	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	10.05.2024	4.06.2024	
7.	Захист магістерської кваліфікаційної роботи	10.06.2024	10.06.2024	

Студент _____ Ратушняк С.С.
(підпис)

Керівник роботи _____ Грицак А.В
(підпис)

АНОТАЦІЯ

УДК 004.56.5(043.2)

Ратушняк С.С. Магістерська кваліфікаційна робота зі спеціальності 125 – «Кібербезпека», освітня програма «Кібербезпека інформаційних технологій та систем». Вінниця: ВНТУ, 2024. – 123 с.

Укр. мовою. Бібліогр.: 41 назв; рис.: 20; табл. 9.

У магістерській кваліфікаційній роботі досліджуються та вдосконалення системи моніторингу та управління IoT-пристроями на основі моделі нульової довіри (Zero Trust) із використанням алгоритму динамічного встановлення довіри.

У першому розділі розглядаються теоретичні засади підвищення захищеності IoT-систем. Описано поняття та класифікацію IoT-пристроїв, основні загрози та вразливості IoT-систем, технології та протоколи управління і моніторингу IoT-пристроями.

Другий розділ присвячено створенню захищеної системи моніторингу та управління IoT-пристроями. Розглянуто особливості вдосконалення моделі нульової довіри та використання алгоритму динамічного встановлення довіри. Запропоновано проектування та розробка алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями.

У третьому розділі наводиться програмна реалізація розробленої системи. Обґрунтовується вибір мови реалізації та середовища розробки, здійснюється безпосередня реалізація системи моніторингу та управління IoT-пристроями та вдосконаленої моделі нульової довіри з використанням алгоритму динамічного встановлення довіри. Проводиться тестування реалізованої системи та надаються висновки.

Четвертий розділ присвячений економічній частині роботи. Оцінюється комерційний потенціал розробленого програмного забезпечення.

Ключові слова: нульова довіра, IoT, динамічного встановлення довіри

ABSTRACT

Ratushnyak S.S. Master's qualification thesis on specialty 125 - "Cyber security", educational program "Cyber security of information technologies and systems". Vinnytsia: VNTU, 2024. - 127 p.

Ukraine language Bibliography: 41 titles; Fig.: 20; table 9.

In the master's qualification work, the system of monitoring and control of IoT devices based on the Zero Trust model with the use of the algorithm of dynamic establishment of trust is investigated and improved.

The first chapter deals with the theoretical principles of improving the security of IoT systems. The concept and classification of IoT devices, the main threats and vulnerabilities of IoT systems, technologies and protocols for managing and monitoring IoT devices are described.

The second section is devoted to the creation of a secure monitoring and management system for IoT devices. Features of the improvement of the zero trust model and the use of the dynamic trust establishment algorithm are considered. The design and development of an algorithm for increasing the security of the IoT device monitoring and control system is proposed. The chapter ends with conclusions.

The third section presents the software implementation of the developed system. The choice of implementation language and development environment is substantiated, the direct implementation of the IoT device monitoring and control system and the improved zero trust model using the dynamic trust establishment algorithm is carried out. Testing of the implemented system is carried out and conclusions are provided.

The fourth chapter is devoted to the economic part of the work. The commercial potential of the developed software is evaluated, the costs of carrying out scientific work and implementing its results, as well as the commercial effects of the implementation, are predicted.

Keywords: zero trust, IoT, dynamic trust establishment

ЗМІСТ

ВСТУП

Актуальність. Розвиток Інтернету речей (IoT) протягом останніх десятиліть суттєво змінив спосіб взаємодії з технологіями у різних сферах життя, включаючи промисловість, охорону здоров'я, транспорт, сільське господарство та побут. За оцінками експертів, кількість підключених до Інтернету пристроїв продовжує стрімко зростати, створюючи нові можливості для збору, аналізу та використання даних. Однак, цей стрімкий розвиток супроводжується низкою серйозних викликів, зокрема з точки зору безпеки та конфіденційності.

Існуючі методи захисту часто виявляються недостатніми для забезпечення належного рівня безпеки в умовах IoT. Це зумовлено специфічними особливостями IoT-пристроїв, такими як обмежені обчислювальні ресурси, різноманітність протоколів зв'язку та висока гетерогенність мереж. Крім того, велика кількість пристроїв і їх широке географічне розташування роблять IoT-системи вразливими до різних типів атак, включаючи перехоплення даних, несанкціонований доступ, DDoS-атаки та багато інших.

Модель нульової довіри (Zero Trust) є перспективним підходом до забезпечення безпеки в сучасних умовах. Вона передбачає, що жоден користувач або пристрій не може бути автоматично довіреним, навіть якщо вони знаходяться всередині периметру мережі.

Особливо актуальним є застосування алгоритму динамічного встановлення довіри в межах моделі нульової довіри. Такий алгоритм дозволяє адаптувати рівень довіри до пристроїв і користувачів в режимі реального часу, враховуючи поточні умови і загрози. Це забезпечує більш гнучкий і ефективний підхід до управління безпекою, що є критично важливим у динамічному середовищі IoT.

У зв'язку з вищезазначеним, дослідження та розробка вдосконаленої моделі нульової довіри з використанням алгоритму динамічного встановлення довіри є надзвичайно актуальним і важливим завданням. Це дозволить значно підвищити рівень захищеності IoT-систем, забезпечуючи їх стабільне і безпечне функціонування у сучасних умовах кіберзагроз.

Мета і задачі дослідження. Метою дослідження є підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри.

Задачами дослідження є:

- аналіз існуючих загроз та вразливостей iot-систем.
- вивчення сучасних технологій та протоколів управління і моніторингу iot-пристроями.
- розгляд принципів моделі нульової довіри (zero trust) та їх застосування в iot.
- розробка алгоритму динамічного встановлення довіри для iot-систем.
- проектування та реалізація системи моніторингу та управління iot-пристроями на основі вдосконаленої моделі нульової довіри.
- тестування та оцінка ефективності запропонованого рішення.

Об’єкт дослідження. Об’єктом дослідження є система моніторингу та управління IoT-пристроями.

Предмет дослідження. Предметом дослідження є модель нульової довіри (Zero Trust) та алгоритм динамічного встановлення довіри для підвищення захищеності IoT-систем.

Новизна роботи. Новизна роботи полягає у розробці вдосконаленої моделі нульової довіри для IoT-систем з використанням алгоритму динамічного встановлення довіри. Запропонований підхід забезпечує адаптивне управління безпекою, що дозволяє швидко реагувати на нові загрози та змінювати умови експлуатації IoT-пристроїв.

Практична цінність. Практична цінність роботи полягає у створенні захищеної системи моніторингу та управління IoT-пристроями, яка може бути впроваджена в різних галузях, таких як промисловість, медицина, транспорт та інші сфери, де використовується IoT. нових, більш захищених систем.

1 ТЕОРЕТИЧНІ ЗАСАДИ ПІДВИЩЕННЯ ЗАХИЩЕНОСТІ СИСТЕМИ МОНІТОРИНГУ ТА УПРАВЛЯННЯ ІОТ ПРИСТРОЯМИ НА ОСНОВІ ВДОСКОНАЛЕНОЇ МОДЕЛІ НУЛЬВОЇ ДОВІРИ (ZERO TRUST) З ВИКОРИСТАННЯМ АЛГОРИТМУ ДИНАМІЧНОГО ВСТАНОВЛЕННЯ ДОВІРИ

У першому розділі магістерської кваліфікаційної роботи також буде проведено аналіз основних викликів та проблем, з якими стикаються системи моніторингу та управління IoT пристроями. Розглянуться сучасні тенденції у розвитку IoT технологій та їх вплив на безпеку та ефективність систем. Також буде надано огляд існуючих методів та стратегій забезпечення безпеки в контексті IoT, зокрема моделей нульової довіри та методів аутентифікації та авторизації.

Крім того, у розділі буде описано постановку задач та мету подальшого дослідження, що спрямоване на підвищення захищеності та ефективності систем управління IoT пристроями.

1.1 Поняття та класифікація IoT пристроїв

Інтернет речей (IoT, Internet of Things) - це концепція, що описує мережу фізичних об'єктів ("речей"), які мають вбудовані технології для взаємодії один з одним та з навколишнім середовищем через Інтернет або інші мережі. Ці об'єкти можуть включати різноманітні пристрої, такі як датчики, актуатори, програмне забезпечення, а також мережеві можливості для збору та обміну даними. Основна ідея IoT полягає в створенні інтелектуальної системи, де кожен об'єкт має можливість автономно функціонувати, реагувати на зовнішні зміни та взаємодіяти з іншими об'єктами [1].

Основні елементи IoT пристроїв:

– сенсори;

– актуатори;

– комунікаційні модулі;

– обчислювальні ресурси;

– інтерфейси.

Розглянемо кожен елемент IoT пристроїв детальніше:

– сенсори;

Функція: сенсори є ключовими компонентами IoT пристроїв, оскільки вони дозволяють збирати дані про фізичний світ. Вони можуть вимірювати різноманітні параметри, такі як температура, вологість, тиск, світло, рух, звукові сигнали та інше.

Приклади: температурні датчики, датчики вологості, світлові сенсори, акселерометри, гіроскопи, звукові датчики.

– актуатори;

Функція: актуатори перетворюють цифрові сигнали в фізичні дії. Вони використовуються для виконання певних завдань у відповідь на сигнали з сенсорів або інших пристроїв [2].

Приклади: електромагнітні клапани, електродвигуни, реле, підігрівачі, світлодіоди.

– комунікаційні модулі;

Функція: ці модулі забезпечують передачу даних між IoT пристроями та мережею. Вони підтримують різноманітні протоколи та стандарти бездротового та дротового зв'язку.

Приклади: Wi-Fi модулі, Bluetooth модулі, Zigbee, LoRa, NB-IoT, Ethernet.

– обчислювальні ресурси;

Функція: включають процесори та пам'ять, які дозволяють пристроям обробляти дані та виконувати різні алгоритми. Вони забезпечують виконання задач у реальному часі.

Приклади: мікроконтролери (наприклад, Arduino), одноплатні комп'ютери (наприклад, Raspberry Pi), спеціалізовані чіпи для AI та машинного навчання (наприклад, NVIDIA Jetson).

– інтерфейси.

Функція: інтерфейси дозволяють взаємодіяти з IoT пристроями та керувати ними. Вони можуть бути як фізичними (кнопки, дисплеї), так і віртуальними (мобільні додатки, веб-інтерфейси).

Приклади: сенсорні екрани, клавіатури, голосові інтерфейси (Alexa, Google Assistant).

IoT пристрої функціонують за рахунок комплексної взаємодії різних компонентів та технологій. Основні принципи роботи IoT пристроїв включають збір даних, обробку даних, передачу даних, аналіз даних та виконання дій. Нижче розглянемо кожен з цих принципів детальніше [3].

Збір даних є першим і ключовим етапом в роботі IoT пристроїв. Він здійснюється за допомогою сенсорів, які вимірюють різні параметри навколишнього середовища або об'єктів.

Типи сенсорів:

– температурні сенсори: вимірюють температуру середовища або об'єктів;

– сенсори вологості: вимірюють рівень вологості повітря або ґрунту;

– світлові сенсори: вимірюють інтенсивність освітлення;

– акселерометри та гіроскопи: вимірюють рух та орієнтацію пристроїв;

– газові сенсори: визначають концентрацію різних газів у повітрі;

– біомедичні сенсори: вимірюють фізіологічні показники, такі як серцевий ритм або рівень глюкози в крові.

Після збору даних сенсорами, вони передаються на обробку. Обробка може відбуватися локально на самому IoT пристрої або в централізованому вузлі.

Локальна обробка (Edge Computing):

– мікроконтролери: прості обчислювальні пристрої, що виконують базові задачі обробки даних;

– одноплатні комп’ютери: потужніші пристрої, які можуть виконувати складніші обчислювальні задачі;

– процесори та чіпи AI: спеціалізовані обчислювальні пристрої для виконання задач штучного інтелекту та машинного навчання.

Переваги локальної обробки:

– зменшення затримок: обробка даних на місці зменшує час реакції;

– зниження навантаження на мережу: менше даних передається через мережу, що знижує трафік.

Після обробки дані передаються до центральної системи або хмарного сервісу для подальшого аналізу та зберігання. Передача даних здійснюється за допомогою різних комунікаційних протоколів та технологій.

Протоколи передачі даних:

– Wi-Fi: використовується для підключення до локальних мереж з високою швидкістю передачі даних;

– Bluetooth: забезпечує передачу даних на коротких відстанях з низьким енергоспоживанням;

– Zigbee: протокол для малопотужних пристроїв з низькою швидкістю передачі даних;

– LoRa: забезпечує передачу даних на великі відстані з низьким енергоспоживанням;

– NB-IoT: технологія для підключення пристроїв до мережі Інтернет з низьким енергоспоживанням.

Методи передачі даних:

– дротові методи: Ethernet, USB;

– бездротові методи: Wi-Fi, Bluetooth, Zigbee, LoRa, NB-IoT.

Дані, зібрані та передані IoT пристроями, підлягають аналізу для отримання корисної інформації та прийняття рішень. Аналіз може проводитися як локально, так і в хмарних сервісах [4].

Методи аналізу даних:

– статистичний аналіз: використовується для виявлення закономірностей та трендів у даних;

– машинне навчання: алгоритми використовуються для побудови моделей, які можуть передбачати майбутні події або класифікувати дані;

– аналіз у реальному часі: дозволяє отримувати інформацію та реагувати на події в режимі реального часу.

Переваги хмарного аналізу:

– масштабованість: хмарні сервіси можуть обробляти великі обсяги даних;

– зберігання даних: можливість зберігати дані протягом тривалого часу;

– інтеграція: можливість інтеграції з іншими сервісами та додатками.

На основі проаналізованих даних приймаються рішення та виконуються відповідні дії. Актуатори, керовані IoT пристроями, виконують фізичні дії, наприклад, включення/вимкнення пристроїв, регулювання параметрів середовища [5].

Приклади виконання дій:

– автоматизація систем освітлення: вмикання або вимикання світла в залежності від рівня освітленості та присутності людей;

– управління кліматичними системами: регулювання температури та вологості на основі зібраних даних;

– моніторинг та управління виробничими процесами: автоматичне коригування параметрів обладнання для оптимізації продуктивності.

Ролі актуаторів:

– перетворення сигналів в дії: актуатори отримують команди від обчислювальних модулів та виконують фізичні дії;

– зворотній зв'язок: забезпечують зворотній зв'язок для системи, повідомляючи про виконання дій або про стан системи.

Інтернет речей (IoT) надає можливість створювати інтелектуальні системи в різних галузях, що дозволяє автоматизувати процеси, підвищувати ефективність, зменшувати витрати та покращувати якість життя. На рисунку 1.1 коротко представлено приклади використання IoT пристроїв.

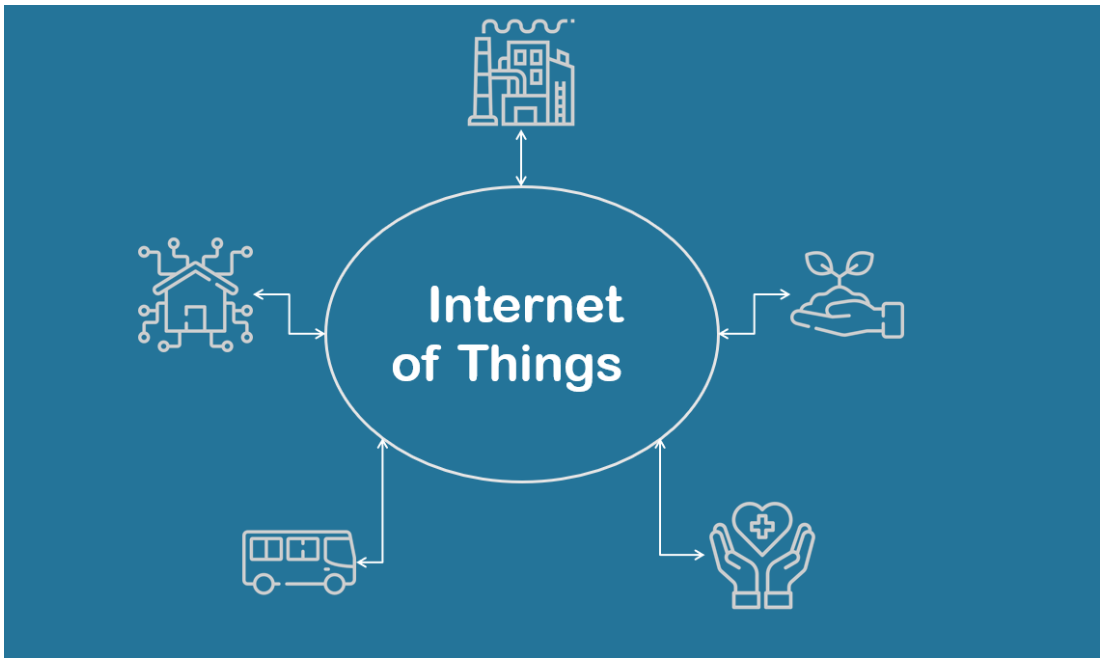


Рисунок 1.1 – Приклади використання IoT пристроїв

Нижче наведено детальні приклади використання IoT пристроїв у різних сферах.

Розумні будинки:

– системи освітлення;

Автоматизація освітлення: Освітлення може автоматично вмикатися і вимикатися на основі часу доби або присутності людей у кімнаті за допомогою датчиків руху.

Регулювання яскравості: Інтенсивність освітлення може бути автоматично регульована залежно від рівня природного світла.

– термостати;

Розумні термостати: Здатні навчатися звичкам мешканців та регулювати температуру відповідно до їхньої присутності та вподобань, що допомагає знизити споживання енергії.

– системи безпеки;

Розумні камери спостереження: Камери з вбудованими сенсорами руху та можливістю відправляти сповіщення на мобільні пристрої у разі виявлення підозрілої активності.

Дверні дзвінки з камерою: Дозволяють бачити та спілкуватися з відвідувачами через смартфон навіть під час відсутності вдома.

– побутова техніка.

Розумні холодильники: Можуть відстежувати запаси продуктів, складати списки покупок та навіть робити замовлення онлайн.

Розумні пральні машини: Можуть оптимізувати використання води та електроенергії, а також надсилати сповіщення про завершення циклу прання.

Промисловий IoT:

– моніторинг стану обладнання;

Вібраційні сенсори: Використовуються для моніторингу стану обладнання та виявлення аномалій, що можуть свідчити про потенційні поломки.

Термальні камери: Дозволяють виявляти перегрівання компонентів, що може запобігти аваріям.

– автоматизація виробничих процесів;

Роботизовані системи: Виконують завдання з високою точністю та швидкістю, зменшуючи потребу у ручній праці.

Системи управління складом: Автоматизовані складські системи використовують RFID технології для відстеження та управління запасами в реальному часі.

– Енергоменеджмент.

Системи моніторингу енергоспоживання: Відстежують використання енергії в реальному часі, що дозволяє виявити неефективне використання ресурсів та зменшити витрати.

Сільське господарство:

– моніторинг стану ґрунту;

Сенсори вологості: Вимірюють рівень вологості ґрунту, що дозволяє оптимізувати зрошення та уникнути пересихання або перенасичення вологою.

Аналіз поживних речовин: Сенсори, що вимірюють вміст поживних речовин у ґрунті, допомагають визначити необхідність внесення добрив.

– системи зрошення;

Автоматизовані системи зрошення: Використовують дані від сенсорів вологості та погодних умов для автоматичного регулювання поливу, що знижує витрати води та підвищує врожайність [6].

– використання дронів.

Моніторинг посівів: Дрони з камерами та сенсорами можуть проводити аерофотозйомку та аналіз стану посівів, виявляючи проблеми на ранніх стадіях.

Розпилення добрив та пестицидів: Автоматизоване розпилення забезпечує рівномірний розподіл речовин, що Підвищує ефективність обробки.

Медицина:

– моніторинг стану пацієнтів;

Носимі пристрої: Фітнес-трекери, смарт-годинники та інші носимі пристрої можуть відстежувати фізіологічні параметри, такі як серцевий ритм, рівень кисню в крові, артеріальний тиск.

Дистанційний моніторинг: Сенсори, що контролюють стан пацієнтів на дому, передають дані лікарям у режимі реального часу, що дозволяє швидко реагувати на зміни стану здоров'я.

– розумні медичні пристрої;

Інсулінові помпи: Автоматично регулюють подачу інсуліну в залежності від рівня глюкози в крові.

Розумні таблетки: Вбудовані сенсори відстежують прийом ліків та передають інформацію лікарям.

– теле-медицина;

Відеоконсультації: Пацієнти можуть отримувати консультації лікарів дистанційно за допомогою відеозв'язку.

Електронні медичні записи: Дані про пацієнтів зберігаються в хмарних сервісах, що полегшує доступ лікарів до історії хвороби.

Транспорт:

– розумні автомобілі;

Автономні транспортні засоби: Використовують сенсори, камери та алгоритми машинного навчання для автономного управління без участі людини.

Системи допомоги водієві (ADAS): Попереджають про перешкоди, контролюють швидкість та смугу руху, що підвищує безпеку на дорогах.

– інтелектуальні транспортні системи;

Управління трафіком: Сенсори та камери відстежують стан доріг та трафік у реальному часі, що дозволяє оптимізувати рух та зменшити затори.

Системи паркування: Розумні паркувальні сенсори допомагають водіям знаходити вільні місця для паркування та оптимізують використання паркувальних зон.

– моніторинг стану транспорту.

Трекінг вантажів: Системи відстеження вантажів дозволяють контролювати їх місцезнаходження та стан під час транспортування.

Діагностика автомобілів: Сенсори в транспортних засобах відстежують технічний стан та повідомляють про необхідність обслуговування або ремонту [7].

Використання IoT пристроїв у різних сферах дозволяє створювати інтелектуальні системи, що підвищують ефективність, знижують витрати та покращують якість життя. Завдяки збиранню, обробці та аналізу даних у реальному часі, IoT відкриває нові можливості для автоматизації, оптимізації процесів та прийняття рішень на основі достовірних даних.

1.2 Основні загрози та вразливості IoT систем

Інтернет речей (IoT) забезпечує значні переваги у багатьох сферах, але також створює нові виклики у сфері безпеки. Вразливості IoT пристроїв та систем можуть призвести до серйозних наслідків, таких як витік конфіденційних даних, фінансові втрати та компрометація фізичної безпеки. Нижче розглянемо основні загрози та вразливості IoT систем.

IoT пристрої мають ряд вразливостей на рівні апаратного забезпечення та мікропрограмного забезпечення, які можуть бути використані зловмисниками для несанкціонованого доступу, маніпуляції даними або збоїв у роботі пристроїв. Нижче детально розглянемо основні вразливості на рівні пристроїв [9].

Багато IoT пристроїв встановлюються в місцях, де вони можуть бути легко доступні для зловмисників. Фізичний доступ до пристроїв дозволяє зловмисникам безпосередньо маніпулювати обладнанням, отримувати доступ до внутрішніх компонентів або навіть повністю викрадати пристрої.

Приклади вразливостей:

– відсутність фізичних засобів захисту, таких як замки або антивандальні корпуси;

– легкий доступ до портів, таких як usb, ethernet або серійні порти, що можуть бути використані для підключення до пристрою та проведення атак.

Багато IoT пристроїв мають слабкі або взагалі відсутні механізми захисту від фізичного злому, що дозволяє зловмисникам отримувати доступ до внутрішніх компонентів та маніпулювати ними.

Приклади вразливостей:

– відсутність захисту мікросхем від прямого доступу та аналізу;

– відсутність механізмів захисту від перепрограмування мікроконтролерів.

Багато IoT пристроїв поставляються з дефолтними або слабкими паролями, які легко вгадати або знайти у відкритих джерелах. Це дозволяє зловмисникам отримувати несанкціонований доступ до пристроїв та їх налаштувань [8].

Приклади вразливостей:

– дефолтні логіни та паролі, що не змінюються користувачами після встановлення пристрою.

– використання простих паролів, таких як "123456" або "password".

Багато IoT пристроїв не отримують регулярних оновлень мікропрограмного забезпечення (firmware), що залишає їх вразливими до відомих вразливостей та експлойтів.

Приклади вразливостей:

– відсутність механізмів автоматичного оновлення мікропрограмного забезпечення.

– відсутність підтримки від виробника після випуску пристрою.

Відсутність механізмів перевірки автентичності та цілісності мікропрограмного забезпечення дозволяє зловмисникам завантажувати на пристрої підроблені або модифіковані мікропрограми.

Приклади вразливостей:

– відсутність цифрових підписів для мікропрограм.

– можливість перепрограмування пристрою без перевірки автентичності програмного забезпечення.

Багато IoT пристроїв використовують незашифровані протоколи для передачі даних, що робить їх вразливими до перехоплення та аналізу зловмисниками.

Приклади вразливостей:

– використання HTTP замість HTTPS для передачі даних через Інтернет;

– відсутність шифрування для локальних з'єднань між пристроями.

Деякі IoT пристрої не мають належних механізмів автентифікації, що дозволяє зловмисникам отримувати несанкціонований доступ до мережевих ресурсів та взаємодіяти з іншими пристроями.

Приклади вразливостей:

– відсутність автентифікації для підключення до пристрою через мережеві інтерфейси;

– відсутність механізмів двофакторної автентифікації.

Багато IoT пристроїв не мають належного захисту від шкідливого програмного забезпечення, що дозволяє зловмисникам інфікувати пристрої та використовувати їх для атак, таких як ботнети або майнінг криптовалют.

Приклади вразливостей:

– відсутність антивірусного програмного забезпечення або засобів виявлення шкідливого ПЗ;

– відсутність механізмів захисту від експлойтів та вразливостей у програмному забезпеченні.

Багато IoT пристроїв зберігають конфіденційну інформацію у незашифрованому вигляді або використовують недостатньо надійні методи захисту, що дозволяє зловмисникам отримувати доступ до цієї інформації.

Приклади вразливостей:

– збереження паролів та конфіденційної інформації у відкритому вигляді у внутрішній пам'яті пристрою;

– використання слабких алгоритмів шифрування або відсутність шифрування взагалі.

Багато виробників IoT пристроїв не дотримуються єдиних стандартів безпеки, що призводить до різноманітних рівнів захищеності пристроїв та ускладнює їх інтеграцію у захищені системи.

Приклади вразливостей:

– використання нестандартизованих протоколів та методів захисту;

– недотримання загальноприйнятих практик безпеки під час розробки та впровадження пристроїв.

Відсутність належного управління безпекою на рівні організації може призвести до того, що навіть захищені пристрої стануть вразливими через неправильне налаштування або відсутність належного моніторингу.

Приклади вразливостей:

– відсутність політик та процедур управління безпекою IoT пристроїв;

– недостатній моніторинг та реагування на інциденти безпеки.

Вразливості на рівні пристроїв є одними з найбільш критичних для забезпечення безпеки IoT систем. Забезпечення фізичного захисту пристроїв, впровадження належних механізмів автентифікації, шифрування даних та регулярне оновлення мікропрограмного забезпечення є ключовими аспектами для мінімізації ризиків. Дотримання стандартів безпеки та належне управління безпекою на рівні організації дозволять значно зменшити вразливості та забезпечити безпечне використання IoT технологій [10].

Мережеві вразливості в системах Інтернету речей (IoT) включають ризики, пов'язані з комунікаціями між пристроями, а також з їх підключенням до Інтернету. Детально розглянемо основні вразливості на рівні мережі:

Багато IoT пристроїв передають дані по мережі без використання шифрування, що робить їх вразливими до перехоплення та аналізу зловмисниками.

Приклади вразливостей:

– використання незашифрованих протоколів, таких як HTTP, для передачі конфіденційних даних;

– відсутність захисту шифрування на рівні бездротових мереж, таких як Wi-Fi або Bluetooth.

Деякі IoT пристрої мають вразливості в їхньому мережевому стеці, які можуть бути використані для виконання атак, таких як витік інформації або відмова в обслуговуванні (DoS).

Приклади вразливостей:

– використання буферних переповнень або інших вразливостей в мережевих протоколах для відмови в обслуговуванні;

– використання вразливостей в мережевих бібліотеках для витоку конфіденційної інформації.

Деякі IoT пристрої підключаються до мережі без відповідних заходів безпеки, що може створювати ризики для всієї мережі.

Приклади вразливостей:

– використання слабких або дефолтних паролів для доступу до мережевих пристроїв;

– відсутність периметральних засобів захисту, таких як брандмауери або інтрафейси інспекції пакетів.

Деякі IoT пристрої використовують застарілі мережеві протоколи або стандарти, які можуть бути вразливими до відомих атак.

Приклади вразливостей:

– використання застарілих версій бездротових протоколів, таких як WEP для Wi-Fi, які мають відомі вразливості;

– використання застарілих версій мережевих протоколів, таких як SNMP або Telnet, які можуть бути вкрай небезпечними через відкриті вразливості.

Деякі IoT пристрої можуть бути компрометовані через атаки, спрямовані безпосередньо на їх мережеві пристрої, такі як маршрутизатори, комутатори або файерволи.

Приклади вразливостей:

– використання вразливостей в програмному забезпеченні мережевих пристроїв для отримання несанкціонованого доступу;

– використання атак, таких як атаки з переповнення буфера, для виконання зловмисних дій на мережевих пристроях.

Вразливості на рівні мережі є серйозною загрозою для безпеки систем IoT. Недостатнє шифрування трафіку, атаки на мережевий стек, недостатні заходи безпеки мережі, використання неактуальних протоколів та стандартів, а також атаки на пристрої мережі можуть призвести до компрометації всієї інфраструктури IoT. Забезпечення належного захисту мережі, включаючи шифрування трафіку, оновлення програмного забезпечення та застосування надійних методів аутентифікації та авторизації, є важливими кроками для мінімізації ризиків і забезпечення безпеки систем IoT [11].

Вразливості на рівні додатків в системах Інтернету речей (IoT) включають в себе ризики, пов'язані з програмним забезпеченням, що використовується для керування та моніторингу пристроїв, а також з інтеграцією з веб-сервісами та хмарними платформами. Розглянемо основні вразливості на рівні додатків:

Багато IoT додатків мають недостатні або слабкі механізми аутентифікації та авторизації, що може призвести до несанкціонованого доступу до функцій та даних.

Приклади вразливостей:

– використання простих або дефолтних паролів для входу до додатків.;

– недостатність перевірки прав доступу до ресурсів та функцій додатків.

Деякі IoT додатки можуть використовувати ненадійні алгоритми шифрування для захисту конфіденційної інформації, що робить їх вразливими до атак на перехоплення та декодування даних.

Приклади вразливостей:

– використання застарілих або слабких шифрувальних алгоритмів, таких як DES або MD5;

– недостатня довжина ключів шифрування, що робить їх вразливими до атак методом перебору.

Деякі IoT додатки можуть бути вразливими до атак, під час яких зловмисник може захопити аутентифікаційні сесії користувачів та отримати несанкціонований доступ до їх облікових записів.

Приклади вразливостей:

– використання ненадійних механізмів сесійного керування, таких як відкритий текстовий ідентифікатор сесії або недостатня захищеність транспортного каналу.

Деякі IoT додатки можуть бути вразливими до атак типу "ін'єкція", під час яких зловмисник може вводити та виконувати шкідливі команди або запити в системі [12].

Приклади вразливостей:

– SQL Injection: зловмисник вводить SQL-запити через веб-інтерфейс, що може призвести до видалення або зміни даних в базі даних.

– XSS (Cross-Site Scripting): зловмисник вводить скрипти JavaScript через веб-інтерфейс, які виконуються у контексті користувача та можуть використовувати його сесію.

Деякі IoT додатки можуть не забезпечувати належного аудитування та моніторингу дій користувачів, що може ускладнити виявлення та реагування на випадки порушення безпеки.

Приклади вразливостей:

– відсутність журналування дій користувачів або адміністраторів системи.

– недостатність механізмів сповіщення адміністраторів про підозрілі або несанкціоновані дії.

Вразливості на рівні додатків можуть становити серйозну загрозу для безпеки систем IoT, дозволяючи зловмисникам отримати несанкціонований доступ до функцій та даних пристроїв. Забезпечення належного рівня аутентифікації та авторизації, використання надійних алгоритмів шифрування, захист від атак типу "захоплення сесії", захист від ін'єкцій та належний моніторинг дій користувачів є важливими кроками для забезпечення безпеки додатків у системах IoT.

Вразливості на рівні даних в системах Інтернету речей (IoT) включають в себе ризики, пов'язані зі збереженням, обробкою та передачею інформації, що збирається та обробляється IoT пристроями. Розглянемо основні вразливості на рівні даних:

Деякі IoT пристрої можуть зберігати конфіденційні дані у ненадійних форматах або без належного шифрування, що робить їх вразливими до несанкціонованого доступу та витоку інформації [13].

Приклади вразливостей:

– зберігання паролів чи інших конфіденційних даних у відкритому або слабкому форматі;

– недостатнє шифрування файлів чи баз даних, що містять конфіденційну інформацію.

Деякі IoT пристрої можуть передавати дані по мережі без використання шифрування або захищених каналів зв'язку, що робить їх вразливими до перехоплення та аналізу зловмисниками.

Приклади вразливостей:

– передача конфіденційних даних через незахищені протоколи, такі як HTTP, без шифрування;

– використання ненадійних механізмів аутентифікації під час передачі даних, таких як передача паролів у відкритому вигляді.

Деякі IoT пристрої можуть мати недоліки в алгоритмах обробки та аналізу даних, що може призвести до атак на витік інформації або зловживання функціями [14].

Приклади вразливостей:

– недостатня фільтрація вхідних даних, що може призвести до атак типу "ін'єкція" або зміни поведінки системи;

– недостатнє обмеження доступу до оброблюваних даних, що може призвести до незаконного доступу до конфіденційної інформації.

Деякі IoT системи можуть мати недоліки в заходах безпеки, що може призвести до витоку конфіденційної інформації через різноманітні атаки.

Приклади вразливостей:

– недостатня захист від атак типу "витік SQL" або "витік XPath", що може призвести до витоку чутливої інформації з баз даних;

– недостатнє шифрування файлів з конфіденційною інформацією, що може призвести до їхнього недоцільного доступу з боку несанкціонованих користувачів.

Багато IoT пристроїв використовують хмарні сервіси для зберігання та обробки даних, і недостатні заходи безпеки в таких сервісах можуть становити загрозу для конфіденційності та цілісності даних [15].

Приклади вразливостей:

– недостатня аутентифікація та авторизація користувачів в хмарних сервісах, що може призвести до несанкціонованого доступу до даних;

– недостатність шифрування даних під час їх передачі та зберігання в хмарних сервісах.

Вразливості на рівні даних можуть становити серйозну загрозу для безпеки систем IoT, дозволяючи зловмисникам отримати доступ до конфіденційної інформації або модифікувати дані. Захист збережених та переданих даних за допомогою шифрування, контроль доступу до даних та аудит системи для виявлення можливих вразливостей є важливими заходами для забезпечення безпеки даних в системах IoT.

Багато компаній не володіють достатньою усвідомленістю ризиків, пов'язаних з використанням IoT пристроїв і систем. Це може призвести до неправильного впровадження заходів безпеки або недооцінки потенційних загроз.

Приклади вразливостей:

– недостатня підготовка персоналу до розуміння потенційних загроз та методів захисту.

– відсутність внутрішньої культури безпеки, що призводить до недооцінки важливості заходів безпеки.

Деякі компанії можуть мати недоліки у процесах управління життєвим циклом безпеки IoT пристроїв, включаючи розробку, випробування, впровадження та підтримку.

Приклади вразливостей:

– недостатня увага до безпеки під час розробки та випробування нових продуктів.

– відсутність процедур оновлення та патчення для застарілих IoT пристроїв.

Деякі компанії можуть мати недостатні або слабкі політики доступу та авторизації, що призводить до неналежного контролю над доступом до систем та даних.

Приклади вразливостей:

– використання слабких паролів або спільних облікових записів для доступу до IoT систем.

– недостатність двофакторної аутентифікації для захисту від несанкціонованого доступу.

Деякі компанії можуть мати недостатність механізмів моніторингу та реагування на інциденти в системах IoT, що призводить до затримки в виявленні та вирішенні потенційних загроз [16].

Приклади вразливостей:

– відсутність систем моніторингу безпеки, які можуть виявляти аномалії та підозрілу активність;

– недостатність планів реагування на інциденти та тренування персоналу щодо їх виконання.

Часто недоліками в безпеці Інтернету речей стають вразливості в постачальницькому ланцюзі, оскільки велика кількість компонентів, програмного забезпечення та послуг можуть бути придбані від сторонніх постачальників [18].

Приклади вразливостей:

– використання компонентів або пристроїв, які мають вбудовані безпекові дефекти або зазнали атак на постачальницькому етапі;

– недостатня перевірка безпеки програмного забезпечення або компонентів, що входять в склад системи, з боку постачальників.

Управлінські та організаційні вразливості можуть становити серйозну загрозу для безпеки систем Інтернету речей.

Захист від цих вразливостей вимагає ретельного планування, впровадження правильних політик та процедур управління безпекою, а також використання надійних постачальників та компонентів.

Організації повинні приділяти достатню увагу цим аспектам для забезпечення високого рівня безпеки їхніх систем IoT.

Таблиця 1.1 "Основні загрози та вразливості IoT систем" класифікує ці ризики за рівнями.

Таблиця 1.1 – Основні загрози та вразливості IoT систем

Рівень	Загрози та вразливості	Приклади
Апаратний та мікропрограмний рівень	- Фізичний доступ до пристроїв - Слабкі механізми захисту від фізичного злому - Дефолтні або слабкі паролі - Відсутність оновлень мікропрограмного забезпечення - Неперевірене мікропрограмне забезпечення	- Відсутність фізичних засобів захисту - Відсутність захисту мікросхем - Дефолтні логіни та паролі - Відсутність механізмів автоматичного оновлення - Відсутність підтримки від виробника
Мережевий рівень	- Незашифровані дані - Вразливості мережевого стеку - Неналежні заходи безпеки мережі - Застарілі протоколи та стандарти - Атаки на мережеві пристрої	- Використання HTTP замість HTTPS - Використання незахищених мереж - Використання слабких або дефолтних паролів - Використання застарілих протоколів - Вразливості в програмному забезпеченні мережевих пристроїв
Рівень додатків	- Недостатні механізми аутентифікації та авторизації - Ненадійні алгоритми шифрування - Атаки захоплення сесій - Атаки типу "ін'єкція" - Неналежне аудитування та моніторинг	- Використання простих або дефолтних паролів - Використання застарілих або слабких алгоритмів - Недостатня захищеність транспортного каналу - SQL Injection, XSS - Відсутність журналування дій

Продовження таблиці 1.1

Рівень даних	- Ненадійне зберігання даних - Незашифровані дані - Недоліки в алгоритмах обробки даних - Недостатні заходи безпеки - Недостатність безпеки хмарних сервісів	- Зберігання паролів у відкритому форматі - Передача даних без шифрування - Недостатня фільтрація вхідних даних - Недостатнє обмеження доступу - Неналежне шифрування даних в хмарних сервісах
Управлінський та організаційний рівень	- Недостатня усвідомленість ризиків - Недоліки в процесах управління життєвим циклом - Недостатні політики доступу та авторизації - Недостатність моніторингу та реагування на інциденти - Вразливості в постачальницькому ланцюзі	- Недостатня підготовка персоналу - Відсутність культури безпеки - Недостатня увага до безпеки під час розробки - Відсутність оновлень для застарілих пристроїв - Використання слабких паролів - Відсутність двофакторної аутентифікації - Відсутність систем моніторингу - Недостатність планів реагування на інциденти

Кожна категорія в таблиці 1.1 містить приклади типових вразливостей. Це допомагає краще зрозуміти характер та потенційні наслідки цих ризиків.

Таблиця 1.1 "Основні загрози та вразливості IoT систем" слугує корисним інструментом для оцінки та пом'якшення ризиків, пов'язаних з IoT. Визнаючи та розуміючи ці вразливості, організації та користувачі можуть вжити заходів для захисту своїх систем та даних.

Системи Інтернету речей (IoT) мають численні загрози та вразливості на різних рівнях, що потребує комплексного підходу до їх захисту. Вразливості можуть виникати на рівні пристроїв через незахищений фізичний доступ, застаріле або ненадійне програмне забезпечення, та недостатній захист збережених даних [19].

На рівні мережі проблеми включають недостатнє шифрування трафіку, вразливості мережевих протоколів та атаки на мережеві пристрої. На рівні додатків часто виникають проблеми зі слабкою аутентифікацією та авторизацією, ненадійними алгоритмами шифрування, вразливостями до ін'єкцій та захоплення сесій, а також недостатнім моніторингом дій користувачів.

Крім того, на рівні даних можуть бути недоліки у захисті збережених та переданих даних, вразливості в обробці даних та витоки конфіденційної інформації. Управлінські та організаційні аспекти також відіграють важливу роль, включаючи недостатню усвідомленість ризиків, недоліки в управлінні життєвим циклом безпеки, слабкі політики доступу та авторизації, недостатній моніторинг та реагування на інциденти, а також проблеми у постачальницькому ланцюгу [20].

Лише інтегрований підхід до безпеки дозволить мінімізувати ризики та захистити систему від потенційних загроз і вразливостей.

1.3 Технології та протоколи управління та моніторингу IoT пристроями

Для ефективного управління та моніторингу IoT пристроїв використовуються різноманітні технології та протоколи, які забезпечують безпечний і надійний обмін даними. Розглянемо ключові технології та протоколи, що застосовуються в цій сфері.

MQTT (Message Queuing Telemetry Transport) – це легкий протокол обміну повідомленнями, розроблений для передачі даних між пристроями в умовах обмежених ресурсів та низької пропускної здатності. Він часто використовується в системах Інтернету речей (IoT) для забезпечення надійного та ефективного

обміну даними між пристроями та серверами. Нижче наведено детальний опис принципів роботи MQTT, його особливостей, переваг та недоліків.

На рисунку 1.2 зображено схему принципів роботи протоколу MQTT (Message Queuing Telemetry Transport), яка ілюструє взаємодію між видавцем, брокером та підписником у моделі "публікація-підписка".

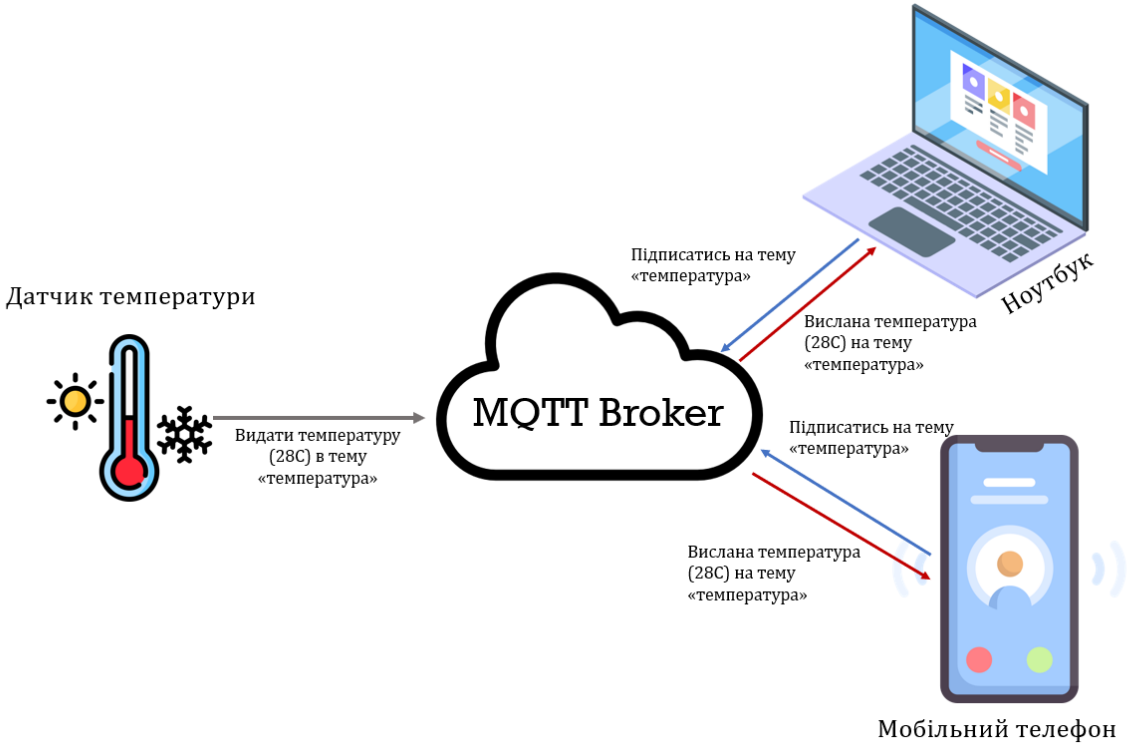


Рисунок 1.2 – Схема принципів роботи протоколу MQTT

Ця схема ілюструє ключові компоненти та процеси роботи протоколу MQTT, показуючи, як дані передаються від видавців до підписників через брокера, забезпечуючи ефективний та надійний обмін інформацією в системах Інтернету речей (IoT) [21].

MQTT базується на моделі "публікація-підписка" (publish-subscribe), що відрізняється від традиційної клієнт-серверної моделі. Ця модель включає три основні компоненти:

– видавець (Publisher): Пристрій або додаток, який генерує та відправляє повідомлення;

– брокер (Broker): Сервер, який приймає повідомлення від видавців та розсилає їх підписникам;

– підписник (Subscriber): Пристрій або додаток, який підписується на отримання повідомлень з певної теми (topic).

Коли видавець надсилає повідомлення, брокер отримує його та розсилає всім підписникам, які підписані на відповідну тему. Це дозволяє розділити завдання відправки та отримання повідомлень між різними пристроями та додатками.

Особливості MQTT

– легкість та ефективність:

MQTT спроектований для роботи в умовах обмежених ресурсів, таких як низька пропускна здатність мережі та обмежена енергоємність пристроїв.

Використання малих за розміром заголовків та ефективне кодування повідомлень.

– рівні якості обслуговування (QoS):

MQTT підтримує три рівні якості обслуговування, що дозволяє контролювати надійність доставки повідомлень:

QoS 0 (At most once): Повідомлення доставляється один раз, без підтвердження доставки.

QoS 1 (At least once): Повідомлення доставляється як мінімум один раз, з підтвердженням доставки.

QoS 2 (Exactly once): Повідомлення доставляється рівно один раз, з підтвердженням та дублюванням доставки.

– Збереження повідомлень (Retained Messages):

MQTT дозволяє зберігати останнє повідомлення для кожної теми, щоб нові підписники могли отримати його відразу після підписки.

– Тривалі сесії (Persistent Sessions):

Підписники можуть відновлювати свої підписки та отримувати накопичені повідомлення після повторного підключення до брокера.

– Відстеження стану клієнтів (Last Will and Testament - LWT):

Видавці можуть залишити "заповіт", що буде відправлений у разі несподіваного відключення.

Переваги MQTT:

– ефективне використання ресурсів: Завдяки малим заголовкам та ефективному кодуванню, MQTT підходить для пристроїв з обмеженими ресурсами та низькою пропускною здатністю мережі;

– гнучкість у налаштуванні надійності: Підтримка різних рівнів якості обслуговування дозволяє налаштувати протокол для досягнення балансу між надійністю та продуктивністю;

– простота реалізації: Протокол має просту структуру, що полегшує його впровадження в різних додатках та пристроях;

– масштабованість: MQTT підходить для широкомасштабних систем з великою кількістю підключених пристроїв завдяки централізованій моделі "публікація-підписка".

Недоліки MQTT:

– залежність від брокера: Надійність системи залежить від роботи брокера, що може стати єдиною точкою відмови;

– безпека: MQTT сам по собі не забезпечує шифрування даних, тому необхідно додатково налаштовувати захищені канали зв'язку (наприклад, за допомогою TLS);

– обмежені можливості для великих повідомлень: Протокол не оптимізований для передачі великих обсягів даних, що може вимагати додаткових механізмів фрагментації та збирання повідомлень.

MQTT є потужним та ефективним протоколом для обміну даними в системах IoT, особливо в умовах обмежених ресурсів. Його легкість, гнучкість у налаштуванні надійності та простота реалізації роблять його популярним вибором для багатьох IoT додатків. Однак для забезпечення належного рівня безпеки та надійності необхідно враховувати потенційні недоліки та використовувати додаткові заходи захисту та резервування.

6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) – це мережевий протокол, розроблений для передачі IPv6 пакетів через мережі з низьким енергоспоживанням, такі як мережі IEEE 802.15.4. Основною метою

6LoWPAN є забезпечення підтримки IPv6 в обмежених середовищах, таких як пристрої Інтернету речей (IoT), де ресурси обмежені, а енергоспоживання має бути мінімальним.

Основні особливості:

– стиснення заголовків IPv6:

Використання стандартного IPv6 у мережах з низькою пропускною здатністю може створювати значні накладні витрати. 6LoWPAN використовує методи стиснення заголовків, щоб зменшити розмір пакетів, що передаються. Це дозволяє ефективно використовувати доступну пропускну здатність.

– фрагментація та реасемблювання:

Пакети IPv6 можуть бути більшими, ніж максимальний розмір кадру, підтримуваний мережами IEEE 802.15.4 (127 байт). 6LoWPAN забезпечує механізми фрагментації великих пакетів на менші частини та їх реасемблювання на стороні приймача [22].

– мобільність та мережеве перепідключення:

6LoWPAN підтримує мобільність вузлів у межах мережі. Пристрій може змінювати своє місцезнаходження в мережі без втрати підключення та з мінімальними затримками.

– низьке енергоспоживання:

6LoWPAN розроблений для роботи з пристроями, що працюють від батарей, забезпечуючи низьке енергоспоживання та довготривалу роботу.

Принципи роботи:

– архітектура мережі:

6LoWPAN мережа зазвичай складається з великої кількості вузлів (сенсорів, актуаторів тощо), які спілкуються між собою та з зовнішнім світом через маршрутизатори.

Головний маршрутизатор (6LoWPAN Border Router) забезпечує зв'язок між 6LoWPAN мережею та зовнішньою IPv6 мережею.

– стиснення заголовків:

6LoWPAN використовує схему стиснення заголовків, відому як "LOWPAN_IPHC" (IP Header Compression), для зменшення накладних витрат. Це дозволяє зменшити 40-байтовий заголовок IPv6 до кількох байтів.

– фрагментація:

Коли пакет IPv6 перевищує максимальний розмір кадру IEEE 802.15.4, він розбивається на кілька фрагментів. Кожен фрагмент має заголовок фрагментації, що включає інформацію про послідовність фрагментів.

– маршрутизація:

6LoWPAN підтримує різні протоколи маршрутизації для ефективного пересилання пакетів через мережу, такі як RPL (Routing Protocol for Low-Power and Lossy Networks).

Переваги 6LoWPAN:

– підтримка IPv6:

6LoWPAN забезпечує безперешкодне використання протоколу IPv6 у мережах з низьким енергоспоживанням, що дозволяє інтегрувати IoT пристрої у глобальну інтернет-інфраструктуру.

– ефективне використання ресурсів:

Завдяки стисненню заголовків та фрагментації, 6LoWPAN оптимізує використання обмежених мережевих ресурсів.

– низьке енергоспоживання:

Протокол розроблений для мінімізації енергоспоживання, що є критичним для пристроїв, що працюють від батарей.

– масштабованість:

6LoWPAN мережі можуть підтримувати велику кількість вузлів, що дозволяє створювати великомасштабні IoT рішення.

– інтероперабельність:

Використання стандартного IPv6 дозволяє легко інтегрувати 6LoWPAN мережі з іншими мережами та інфраструктурою Інтернету.

Недоліки 6LoWPAN

– обмеження пропускної здатності:

Мережі IEEE 802.15.4 мають обмежену пропускну здатність, що може стати вузьким місцем для деяких додатків.

– складність реалізації:

Стиснення заголовків та фрагментація додають складність до реалізації протоколу, що може вимагати додаткових зусиль для розробників.

– залежність від фізичного середовища:

6LoWPAN залежить від характеристик бездротового середовища, що може вплинути на надійність передачі даних у середовищах з високим рівнем перешкод.

6LoWPAN є потужним рішенням для інтеграції пристроїв з низьким енергоспоживанням у глобальну IPv6 інфраструктуру. Він забезпечує ефективне використання мережевих ресурсів, низьке енергоспоживання та підтримку масштабованості, що робить його ідеальним для додатків Інтернету речей. Однак, для досягнення оптимальної продуктивності та надійності, необхідно враховувати обмеження пропускної здатності та складність реалізації протоколу [23].

CoAP (Constrained Application Protocol) – це спеціально розроблений для пристроїв з обмеженими ресурсами протокол прикладного рівня, який забезпечує обмін даними в умовах низької пропускної здатності та обмежених енергетичних ресурсів. Протокол CoAP розроблений Інженерною групою Інтернету (IETF) і описаний у RFC 7252. Основною метою CoAP є забезпечення простого, але ефективного способу для взаємодії IoT пристроїв з серверами та іншими пристроями в мережі.

Основні особливості:

– простота та легкість:

CoAP розроблений таким чином, щоб бути легким і простим у реалізації, що робить його ідеальним для пристроїв з обмеженими обчислювальними можливостями та пам'яттю.

– побудова на основі REST:

Протокол CoAP використовує архітектурний стиль REST (Representational State Transfer), подібно до HTTP, що спрощує взаємодію між клієнтами та серверами. Кожен ресурс ідентифікується URI.

– надійність передачі:

CoAP використовує протокол UDP (User Datagram Protocol) для передачі даних, але додає механізми надійності, такі як підтвердження доставки (ACK) та повторні спроби передачі.

– підтримка мультикасту:

CoAP підтримує мультикаст, що дозволяє одночасно передавати повідомлення кільком пристроям, зменшуючи навантаження на мережу.

– стиснення заголовків:

Заголовки CoAP дуже малі та можуть бути додатково стиснуті для зменшення розміру повідомлень, що знижує накладні витрати при передачі даних.

– асинхронні повідомлення:

CoAP підтримує асинхронний обмін повідомленнями, дозволяючи клієнтам надсилати запити та отримувати відповіді з затримкою, що важливо для пристроїв з обмеженими ресурсами [24].

На рисунку 1.3 зображено схему принципів роботи протоколу CoAP (Constrained Application Protocol), яка ілюструє взаємодію між клієнтом та сервером у середовищах з обмеженими ресурсами.

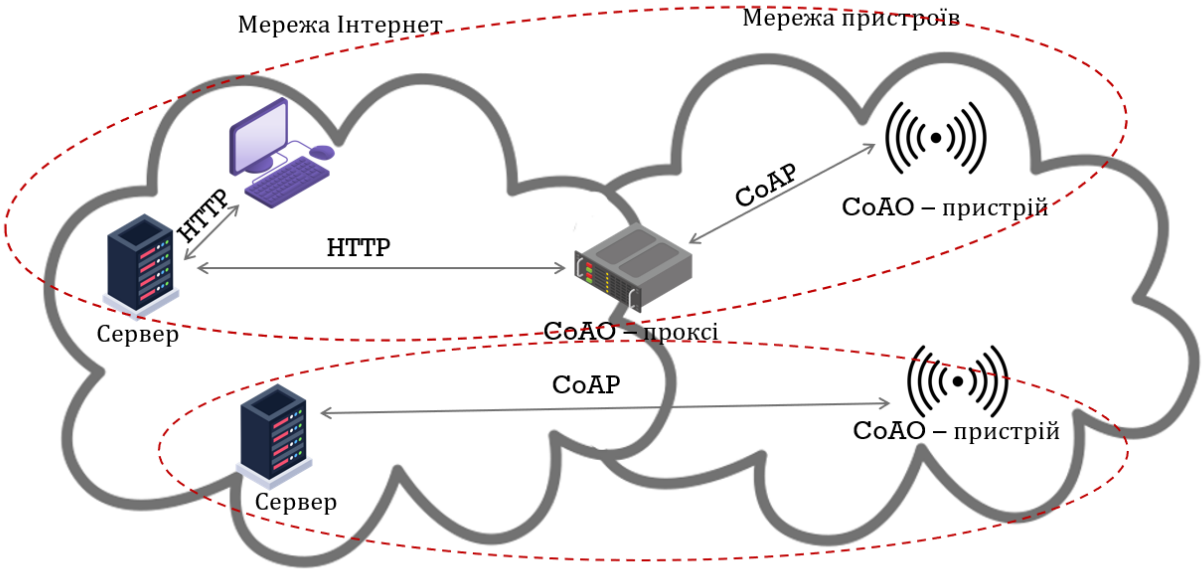


Рисунок 1.3 – Схема принципів роботи протоколу CoAP

Ця схема демонструє ефективність та надійність обміну даними між клієнтом та сервером у мережах з обмеженими ресурсами, використовуючи легкий протокол CoAP.

Принципи роботи:

– методи CoAP:

CoAP підтримує чотири основні методи: GET, POST, PUT та DELETE, аналогічно до HTTP. Ці методи використовуються для взаємодії з ресурсами на сервері.

– повідомлення CoAP:

Повідомлення CoAP складаються з заголовка та, за потреби, корисного навантаження. Заголовок включає тип повідомлення (Confirmable, Non-Confirmable, Acknowledgement, Reset), код запиту або відповіді, ідентифікатор повідомлення та опції.

– типи повідомлень:

Confirmable (CON): Повідомлення, яке потребує підтвердження доставки. Якщо отримувач приймає повідомлення, він відправляє підтвердження (ACK).

Non-Confirmable (NON): Повідомлення, яке не потребує підтвердження доставки. Використовується для менш критичних даних.

Acknowledgement (ACK): Підтвердження прийому повідомлення Confirmable.

Reset (RST): Повідомлення, яке сигналізує про те, що отримане повідомлення не може бути оброблене.

– адресація та маршрутизація:

CoAP використовує URI для ідентифікації ресурсів та підтримує як унітарну, так і мультимедійні адресації, що дозволяє ефективно керувати ресурсами та передавати дані декільком пристроям одночасно.

Переваги CoAP:

– легкість реалізації:

Низькі вимоги до пам'яті та обчислювальних ресурсів роблять CoAP ідеальним для впровадження на обмежених пристроях.

– ефективність:

Стиснення заголовків та підтримка UDP зменшують накладні витрати та покращують продуктивність в умовах обмеженої пропускної здатності.

– надійність:

Механізми підтвердження доставки та повторних спроб передачі забезпечують надійність у нестабільних мережевих середовищах.

– інтероперабельність:

Використання архітектурного стилю REST дозволяє легко інтегрувати CoAP з існуючими веб-технологіями та протоколами.

Недоліки CoAP:

– обмеження безпеки:

Хоча CoAP підтримує DTLS (Datagram Transport Layer Security) для забезпечення безпеки, реалізація на обмежених пристроях може бути складною та вимагати додаткових ресурсів.

– залежність від udp:

Використання UDP як транспортного протоколу може призводити до втрат пакетів у нестабільних мережах, хоча механізми повторної передачі можуть зменшити цей ризик.

– обмежена підтримка складних запитів:

CoAP не підтримує складні запити та функціональні можливості, які доступні в HTTP, що може бути обмеженням для деяких додатків.

CoAP є ефективним та легким протоколом для обміну даними між пристроями з обмеженими ресурсами в умовах низької пропускної здатності. Його простота, ефективність та використання архітектурного стилю REST роблять його ідеальним вибором для додатків Інтернету речей (IoT). Однак для забезпечення безпеки та надійності необхідно враховувати можливі обмеження та використовувати додаткові механізми захисту та управління передачею даних.

Zigbee – це бездротовий протокол передачі даних, розроблений для пристроїв з низьким енергоспоживанням і обмеженими ресурсами. Протокол Zigbee створений для застосування в мережах з низькою швидкістю передачі даних та коротким радіусом дії, зокрема в системах Інтернету речей (IoT). Zigbee розроблений Альянсом Zigbee (Zigbee Alliance) і базується на стандарті IEEE 802.15.4 [25].

Основні особливості

– низьке енергоспоживання;

Zigbee пристрої споживають дуже мало енергії, що дозволяє їм працювати від батарей протягом тривалого часу (до кількох років).

– сітчаста топологія (Mesh Network);

Zigbee підтримує сітчасту топологію мережі, де кожен вузол може передавати дані від інших вузлів, підвищуючи надійність та покриття мережі.

– Масштабованість;

Zigbee мережі можуть включати до 65 000 вузлів, що робить їх придатними для великомасштабних застосувань.

– низька швидкість передачі даних;

Протокол забезпечує швидкість передачі даних до 250 кбіт/с, що достатньо для багатьох IoT застосувань, таких як моніторинг та управління пристроями.

– безпека.

Zigbee включає механізми безпеки, такі як 128-бітове шифрування AES, автентифікація та цілісність даних, забезпечуючи захист мережі та даних від несанкціонованого доступу.

Принципи роботи:

– топологія мережі:

Zigbee підтримує різні типи топологій: зірка, дерево і сітка (mesh). Найпоширенішою є сітчаста топологія, яка забезпечує високу надійність завдяки можливості маршрутизації даних через кілька вузлів.

– типи пристроїв:

Координатор (Coordinator): Основний пристрій мережі, який ініціює мережу та забезпечує її управління.

Маршрутизатор (Router): Пристрій, який передає дані між вузлами та допомагає розширювати покриття мережі.

Кінцевий пристрій (End Device): Пристрій, який виконує кінцеві функції (наприклад, сенсори) і не передає дані від інших вузлів.

– маршрутизація:

Zigbee використовує алгоритми маршрутизації, такі як AODV (Ad hoc On-Demand Distance Vector), для визначення оптимальних маршрутів передачі даних у мережі.

– механізми енергозбереження:

Zigbee пристрої можуть працювати в режимі сну, знижуючи енергоспоживання, коли вони не передають дані. Це дозволяє значно подовжити тривалість роботи від батарей.

Переваги Zigbee

– енергоефективність;

Завдяки низькому енергоспоживанню, Zigbee пристрої можуть працювати від батарей протягом тривалого часу, що спроможності.

– обмежена дальність;

Zigbee має обмежену дальність дії (до 100 метрів на відкритому просторі), що може вимагати використання додаткових маршрутизаторів для покриття великих площ.

– складність налаштування;

Хоча Zigbee підтримує різні топології, налаштування та управління великомасштабними мережами можуть бути складними, особливо для користувачів без спеціалізованих знань.

– конкуренція на ринку.

Zigbee конкурує з іншими протоколами, такими як Z-Wave, Bluetooth Low Energy (BLE) та Thread, що може обмежити його прийняття в деяких галузях.

Zigbee є потужним та ефективним протоколом для побудови мереж IoT пристроїв, які потребують низького енергоспоживання та високої надійності. Його підтримка сітчастої топології, масштабованість та вбудовані механізми безпеки роблять його ідеальним для застосувань у системах автоматизації будівель, розумних містах, промислових системах та домашній автоматизації. Однак, обмеження щодо швидкості передачі даних та дальності дії можуть вимагати додаткових рішень для задоволення специфічних потреб [26].

1.4 Принципи моделі нульової довіри (Zero Trust)

Модель нульової довіри (Zero Trust) – це концепція кібербезпеки, яка базується на принципі "ніколи не довіряй, завжди перевіряй". Вона передбачає, що жоден користувач або пристрій, незалежно від того, чи знаходиться він всередині чи за межами мережі, не може бути автоматично визнаний безпечним. Кожна спроба доступу до ресурсів повинна бути автентифікована, авторизована та перевірена на відповідність політикам безпеки [27].

Основні принципи моделі нульової довіри:

– перевірка користувачів та пристроїв (Verify and Authenticate):

В кожному випадку доступу необхідно перевіряти автентичність користувачів і пристроїв. Це включає двофакторну аутентифікацію (2FA),

багатофакторну аутентифікацію (MFA), біометричну аутентифікацію та інші механізми перевірки.

– обмеження доступу (Least Privilege):

Користувачам та пристроям надається лише той мінімальний рівень доступу, який необхідний для виконання їх завдань. Це зменшує ризики, пов'язані з несанкціонованим доступом до критичних даних та систем.

– мікросегментація (Micro-Segmentation):

Мережа розділяється на ізольовані сегменти для обмеження можливості переміщення зловмисників у разі компрометації однієї частини мережі. Це досягається за допомогою мережевих політик, брандмауерів та інших засобів контролю доступу.

– постійний моніторинг та аналіз (Continuous Monitoring and Analysis):

Вся активність у мережі постійно моніториться і аналізується на предмет аномалій та можливих загроз. Це включає використання систем виявлення вторгнень (IDS), систем запобігання вторгнень (IPS) та інших засобів моніторингу.

– контроль доступу до даних (Data Access Control):

Захист даних реалізується шляхом шифрування, управління правами доступу та політиками безпеки, які визначають, хто і які дані може переглядати або змінювати.

– контроль безпеки кінцевих точок (Endpoint Security Controls):

Забезпечення безпеки кінцевих пристроїв (комп'ютери, мобільні пристрої, IoT пристрої) через антивірусні програми, засоби управління мобільними пристроями (MDM) та інші механізми захисту.

– захист робочих навантажень (Workload Security):

Захист хмарних сервісів, віртуальних машин та інших робочих навантажень шляхом використання відповідних інструментів безпеки, таких як брандмауери для веб-додатків (WAF), системи захисту контейнерів та інших засобів.

Основні компоненти Zero Trust архітектури

– ідентифікація та доступ;

Централізовані системи управління ідентифікацією та доступом (IAM) забезпечують контроль і управління правами доступу користувачів і пристроїв.

– сегментація мережі;

Використання технологій, таких як віртуальні локальні мережі (VLAN), програмно визначені мережі (SDN) та інші засоби для сегментації мережі та обмеження переміщення всередині мережі.

– політики безпеки;

Чітко визначені політики безпеки, що регулюють доступ до ресурсів та поведінку користувачів і пристроїв у мережі.

– моніторинг та аналітика;

Інструменти для моніторингу та аналізу мережевого трафіку, виявлення аномалій та реагування на інциденти безпеки.

– захист даних.

Технології шифрування, управління ключами та інші засоби для забезпечення конфіденційності та цілісності даних.

Переваги моделі нульової довіри:

– підвищена безпека:

Завдяки постійному контролю доступу та перевірці користувачів і пристроїв, модель Zero Trust значно підвищує рівень захищеності організації.

– зменшення ризиків:

Обмеження доступу до ресурсів за принципом найменших привілеїв і мікросегментація зменшують можливості для зловмисників рухатися всередині мережі та доступати до чутливих даних.

– гнучкість:

Модель Zero Trust адаптується до змінних умов роботи, включаючи віддалену роботу та використання хмарних сервісів, забезпечуючи захист у будь-якому середовищі.

– прозорість:

Постійний моніторинг і аналіз активності забезпечують високий рівень прозорості та дають можливість швидко виявляти та реагувати на загрози.

Недоліки та виклики впровадження:

– складність впровадження:

Впровадження моделі Zero Trust вимагає значних змін в архітектурі мережі та системах безпеки, що може бути складним і витратним процесом.

– необхідність навчання:

Потрібне навчання персоналу для розуміння нових політик та процедур безпеки, що може вимагати додаткових ресурсів і часу.

– інтеграція з існуючими системами:

Інтеграція нових рішень безпеки з існуючими системами може бути складною, особливо в умовах великої кількості різнорідних технологій та платформ.

Модель нульової довіри (Zero Trust) пропонує сучасний підхід до кібербезпеки, який значно підвищує рівень захисту організацій від сучасних загроз. Завдяки принципу "ніколи не довіряй, завжди перевіряй", ця модель забезпечує більш надійний контроль доступу до ресурсів і даних, зменшуючи ризики витоків та кібератак. Хоча впровадження моделі Zero Trust може бути складним та вимагати значних ресурсів, її переваги

1.5 Аналіз існуючих моделей та алгоритмів забезпечення довіри в IoT

Забезпечення довіри в IoT системах є надзвичайно важливим для підтримки безпеки та надійності таких систем. Зростаюча кількість підключених пристроїв підвищує вимоги до безпеки, і для їх задоволення використовуються різні моделі та алгоритми забезпечення довіри. Цей розділ розглядає основні моделі та алгоритми, що застосовуються для забезпечення довіри в IoT, аналізує їх переваги та недоліки, а також надає порівняльний аналіз для визначення найбільш ефективних підходів [29].

Централізована модель довіри передбачає використання одного або кількох центральних серверів для автентифікації і авторизації пристроїв у мережі

IoT. Центральний сервер або група серверів виконує функції довіреного посередника, що дозволяє здійснювати управління доступом до ресурсів мережі.

Переваги:

– простота управління: централізований контроль дозволяє легко налаштовувати і управляти політиками безпеки;

– централізоване оновлення: полегшує оновлення системи безпеки і впровадження нових політик.

Недоліки:

– одиничні точки відмови: центральний сервер стає критичною точкою, відмова якої може порушити роботу всієї системи;

– масштабованість: обмежена можливість масштабування через навантаження на центральний сервер.

Децентралізована модель довіри базується на розподіленні функцій довіри серед кількох вузлів мережі без централізованого контролю. Кожен пристрій або вузол у системі може виконувати перевірку автентичності інших пристроїв, що підвищує стійкість до відмов [28].

Переваги:

– стійкість до відмов: відсутність централізованої точки відмови робить систему більш стійкою;

– масштабованість: легше масштабується за рахунок розподілення навантаження між кількома вузлами.

Недоліки:

– складність управління: більш складна координація і управління, що вимагає ефективних протоколів взаємодії між вузлами;

– затримка: можливі затримки в обробці запитів через необхідність взаємодії між багатьма вузлами.

Ієрархічна модель довіри передбачає структуру, де існують рівні довіри, що включають центральні вузли, які контролюють підлеглі вузли. Така структура дозволяє комбінувати переваги централізованої та децентралізованої моделей.

Переваги:

– баланс між централізацією та децентралізацією: дозволяє зберігати централізований контроль на вищих рівнях при децентралізації на нижчих рівнях;

– гнучкість: можна адаптувати структуру в залежності від конкретних вимог і умов.

Недоліки:

– вузькі місця: можливі проблеми на рівнях з великою кількістю підлеглих вузлів;

– складність налаштування: потребує ретельного налаштування і управління для забезпечення ефективності і безпеки.

PKI (Public Key Infrastructure):

– опис: Використовує криптографію з відкритими ключами для автентифікації пристроїв. Центральний сертифікаційний центр (CA) видає цифрові сертифікати;

– переваги: Висока безпека завдяки використанню асиметричної криптографії. Підтримує масштабованість;

– недоліки: Складність управління ключами і сертифікатами. Залежність від центрального сертифікаційного центру.

Symmetric Key Cryptography:

– опис: Використовує один секретний ключ для шифрування та дешифрування даних між двома сторонами;

– переваги: Висока швидкість обробки та низькі вимоги до обчислювальних ресурсів;

– недоліки: Проблеми з безпечним обміном ключами, менш безпечний у порівнянні з асиметричними алгоритмами через можливість компрометації секретного ключа.

Role-Based Access Control (RBAC):

– опис: Контроль доступу на основі ролей, призначених користувачам або пристроям. Кожна роль має певні права і дозволи;

– переваги: Простота у визначенні прав доступу, легкість управління при великих системах;

– недоліки: Обмежена гнучкість, складність в налаштуванні для динамічних середовищ.

Attribute-Based Access Control (ABAC):

– опис: Використовує атрибути (характеристики) користувачів або пристроїв для прийняття рішень щодо доступу. Наприклад, атрибути можуть включати час, місце розташування, тип пристрою тощо;

– переваги: Висока гнучкість, можливість динамічного адаптування політик доступу;

– недоліки: Складність у визначенні та управлінні атрибутами, потребує ретельного налаштування.

Hash Functions (Хеш-функції):

– опис: Використовують криптографічні хеш-функції, такі як SHA-256, для перевірки цілісності даних. Хеш-значення обчислюється для даних і використовується для їх верифікації;

– переваги: Висока швидкість обробки, забезпечення цілісності даних;

– недоліки: Не забезпечують конфіденційності даних.

Message Authentication Code (MAC):

– опис: Використовує секретний ключ для генерації коду автентифікації повідомлень, що підтверджує цілісність та автентичність даних;

– переваги: Забезпечують і цілісність, і автентифікацію повідомлень;

– недоліки: Вимагають управління секретними ключами.

Advanced Encryption Standard (AES):

– опис: Симетричний алгоритм шифрування, який є стандартом для захисту конфіденційних даних. Використовує різні довжини ключів (128, 192, 256 біт);

– переваги: Висока безпека, швидкість обробки, ефективність у використанні ресурсів;

– недоліки: Необхідність управління секретними ключами.

Elliptic Curve Cryptography (ECC):

– опис: Асиметричний алгоритм, що використовує математичні властивості еліптичних кривих для шифрування та підпису даних;

– переваги: Висока безпека при меншій довжині ключа, що робить його ефективним для пристроїв з обмеженими ресурсами;

– недоліки: Складність реалізації та управління ключами.

Проведення порівняльного аналізу моделей та алгоритмів забезпечення довіри в IoT є важливим етапом для розуміння їхніх переваг і недоліків. Це дозволяє обрати найбільш підходящий підхід для конкретних умов і вимог, що, в свою чергу, сприяє підвищенню захищеності IoT систем. Наведені нижче таблиця 1.2 та таблиця 1.3 допоможуть уявити ключові характеристики та особливості кожної з розглянутих моделей і алгоритмів [30].

Таблиця 1.2 – Порівняльна характеристика моделей довіри в IoT

Характеристика	Централізована модель	Децентралізована модель	Ієрархічна модель
Безпека	Висока, але ризик одиничної точки відмови	Висока стійкість до відмов	Висока, компроміс між централізацією та децентралізацією
Масштабованість	Обмежена через навантаження на центральний сервер	Висока, додавання нових вузлів не створює проблем	Краще, ніж у централізованій, але залежить від кількості рівнів
Управління	Простота управління, але складність у масштабуванні	Складність координації між вузлами	Гнучкість у налаштуванні, але складніше управління рівнями

Продовження таблиці 1.2

Ризик одиничних точок відмов	Високий	Низький	Можливий, залежить від структури

Аналіз моделей довіри показує, що централізована модель забезпечує високу безпеку, але має обмежену масштабованість і високий ризик одиничних точок відмов. Децентралізована модель пропонує кращу масштабованість і стійкість до відмов, але потребує складнішого управління. Ієрархічна модель є компромісом між централізацією та децентралізацією, поєднуючи їхні переваги та долаючи деякі недоліки.

Таблиця 1.3 – Порівняльна характеристика алгоритмів забезпечення довіри в IoT

Алгоритм	Переваги	Недоліки
PKI (Public Key Infrastructure)	Висока безпека, можливість інтеграції з іншими системами	Складність управління ключами
Symmetric Key Cryptography	Швидкість, ефективність шифрування	Проблеми з безпечним обміном ключами
Role-Based Access Control (RBAC)	Простота управління, чіткість прав доступу	Обмежена гнучкість, складність зміни прав доступу
Attribute-Based Access Control (ABAC)	Гнучкість, контекстуальність	Складність управління атрибутами
Hash Functions (Хеш-функції)	Простота, ефективність	Не забезпечує конфіденційність даних
Message Authentication Code (MAC)	Цілісність та автентичність даних	Потребує управління секретними ключами

Продовження таблиці 1.3

Advanced Encryption Standard (AES)	Конфіденційність даних, ефективність	Не використовується безпосередньо для автентифікації
Elliptic Curve Cryptography (ECC)	Висока безпека, ефективність	Складність управління ключами

Порівняльний аналіз алгоритмів забезпечення довіри показує, що кожен з них має свої переваги та недоліки. Наприклад, PKI забезпечує високу безпеку, але потребує складного управління ключами. Симетричне шифрування швидке та ефективне, але стикається з проблемами обміну ключами. RBAC забезпечує простоту управління правами доступу, але має обмежену гнучкість. ABAC пропонує високу гнучкість та контекстуальність, але потребує складного управління атрибутами.

Аналіз існуючих моделей та алгоритмів забезпечення довіри в IoT показує, що кожен підхід має свої унікальні переваги та обмеження. Централізовані моделі пропонують високу безпеку, але можуть бути вразливими до одиничних точок відмов і мають обмежену масштабованість. Децентралізовані моделі забезпечують кращу масштабованість і стійкість до відмов, але потребують складнішого управління. Ієрархічні моделі поєднують елементи обох підходів, пропонуючи компроміс між централізацією та децентралізацією [31].

Алгоритми забезпечення довіри, такі як PKI, симетричне шифрування, RBAC, ABAC, хеш-функції, MAC, AES та ECC, мають різні сильні та слабкі сторони. Вибір конкретного алгоритму залежить від вимог до безпеки, продуктивності та управління ключами.

Загалом, для ефективного забезпечення довіри в IoT системах може бути необхідне використання комбінації різних моделей та алгоритмів, щоб досягти оптимального балансу між безпекою та ефективністю. Це дозволить створити надійні та масштабовані IoT системи, здатні ефективно функціонувати в умовах зростаючої складності та кількості підключених пристроїв.

1.6 Висновки та постановка задачі

У першому розділі магістерської кваліфікаційної роботи було проведено детальний аналіз поняття IoT пристроїв та їх класифікації. Розглянуто різноманітні загрози та вразливості IoT систем на різних рівнях, включаючи пристрої, мережі, додатки, дані та управління. Досліджено технології та протоколи, що використовуються для управління та моніторингу IoT пристроїв, такі як MQTT, CoAP, 6LoWPAN та Zigbee.

Вивчено принципи моделі нульової довіри (Zero Trust) та її застосування для підвищення безпеки IoT систем. Проведено аналіз існуючих моделей та алгоритмів забезпечення довіри, включаючи порівняння централізованих, децентралізованих та ієрархічних підходів.

Отже, на основі проведеного дослідження та аналізу було визначено ряд завдань для подальшої розробки:

– здійснити вдосконалення моделі нульової довіри

– здійснити проектування алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями

– здійснити розробку алгоритму підвищення захищеності на основі моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри

– здійснити програмну реалізацію та тестування

2 СТВОРЕННЯ ЗАХИЩЕНОЇ СИСТЕМИ МОНІТОРИНГУ ТА УПРАВЛІННЯ ІОТ-ПРИСТРОЯМИ НА ОСНОВІ ВДОСКОНАЛЕНОЇ МОДЕЛІ НУЛЬОВОЇ ДОВІРИ (ZERO TRUST) З ВИКОРИСТАННЯМ АЛГОРИТМУ ДИНАМІЧНОГО ВСТАНОВЛЕННЯ ДОВІРИ

В даному розіділі будуть розглянутів особливості вдосконалення моделі нульової довіри (Zero Trust) та особливості використанням алгоритму динамічного встановлення довіри, після чого буде здійснено проектування алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями та розробка алгоритму підвищення захищеності на основі моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри.

2.1 Особливості вдосконалення моделі нульової довіри (Zero Trust)

Вдосконалення моделі Zero Trust включає кілька ключових аспектів, які забезпечують підвищену захищеність і адаптивність в умовах сучасних кіберзагроз. Нижче наведено основні напрями вдосконалення моделі Zero Trust.

Принципи Zero Trust:

– відсутність апріорної довіри: кожен доступ до ресурсів повинен бути перевірений незалежно від того, знаходиться користувач або пристрій всередині чи поза межами мережі.

– перевірка кожного доступу: кожна спроба доступу до системи повинна бути ретельно перевірена на основі поточних правил та політик.

– мінімізація привілеїв: користувачам та пристроям повинні надаватися тільки ті привілеї, які необхідні для виконання їхніх задач (принцип найменших привілеїв).

Удосконалення моделі нульової довіри передбачає впровадження сучасних методів аутентифікації та моніторингу.

Багатофакторна аутентифікація (MFA): Включення додаткових факторів аутентифікації, таких як токени, біометричні дані, SMS-коди.

Біометричні методи: Використання біометричних даних, таких як відбитки пальців, розпізнавання обличчя, голосова аутентифікація, для підвищення рівня безпеки.

Контекстуальна аутентифікація: Оцінка додаткових параметрів, таких як місцезнаходження, час доступу, тип пристрою, для визначення рівня довіри.

Аналіз поведінки користувачів (UBA) та пристроїв (UEBA): Використання алгоритмів машинного навчання для виявлення аномальної поведінки, яка може свідчити про загрозу.

Постійний моніторинг: Безперервне спостереження за активністю в мережі, аналіз логів, подій та транзакцій у реальному часі.

Інтелектуальні системи сповіщення: Впровадження систем, що автоматично повідомляють адміністраторів про підозрілу активність або потенційні загрози.

Удосконалення моделі нульової довіри (Zero Trust) вимагає впровадження сучасних технологій, які забезпечують високу адаптивність і ефективність захисту від кіберзагроз. Основні технологічні аспекти включають використання штучного інтелекту та машинного навчання, інтеграцію з хмарними сервісами та сегментацію мережі.

Інтеграція політик Zero Trust з хмарними сервісами забезпечує захист даних та додатків, що зберігаються та обробляються в хмарі. Це включає використання шифрування, багатофакторної аутентифікації та контролю доступу на основі ролей.

Гібридні та мультихмарні середовища: Політики Zero Trust повинні бути адаптовані для роботи в гібридних та мультихмарних середовищах, забезпечуючи єдиний рівень захисту незалежно від розташування даних [32].

Управління ідентифікацією та доступом (IAM) є критичним компонентом вдосконаленої моделі нульової довіри (Zero Trust), оскільки воно забезпечує контроль над доступом користувачів та пристроїв до системних ресурсів. Основні аспекти IAM включають централізоване управління, контроль

привілеїв, багатофакторну аутентифікацію та моніторинг активності користувачів.

Централізоване управління є основою ефективної системи IAM. Воно передбачає одноріневу або багаторівневу ієрархію, де кожен рівень відповідає за управління доступом до певних ресурсів. Це забезпечує єдину точку керування, де адміністратори можуть надавати, змінювати та відкликати права доступу користувачам та пристроям відповідно до принципів Zero Trust.

Принцип найменших привілеїв є основоположним для безпеки в рамках моделі Zero Trust. Система IAM забезпечує відповідність цьому принципу, надаючи користувачам лише ті привілеї, які є необхідними для виконання їхніх функцій. Це зменшує ризик компрометації в разі, якщо аккаунт користувача або пристрою стане жертвою атаки.

Багатофакторна аутентифікація є однією з ключових технік забезпечення безпеки при доступі до системних ресурсів. Вона вимагає введення додаткових параметрів, окрім традиційного логіну та пароля, таких як токени, біометричні дані, SMS-коди тощо. Це значно ускладнює можливість несанкціонованого доступу навіть у випадку, якщо логін та пароль були скомпрометовані.

Моніторинг активності користувачів є важливою складовою для виявлення аномальної поведінки, що може свідчити про можливу загрозу. Системи IAM здатні аналізувати та виявляти незвичайні підходи до вхідних даних, що дає можливість вчасно втручатися і запобігати можливим загрозам.

Інтеграція політик Zero Trust з хмарними інфраструктурами є необхідною для забезпечення захисту даних із зберіганням та обробки в хмарному середовищі. Це включає в себе застосування шифрування, багатофакторної аутентифікації та контролю доступу на основі ролей.

Управління ідентифікацією та доступом також повинне бути адаптоване до роботи в гібридних та мультихмарних середовищах. Це вимагає створення єдиної системи управління, яка забезпечує безпеку та надійність доступу до ресурсів незалежно від їх місцезнаходження [33].

Удосконалення моделі нульової довіри є безперервним процесом, що вимагає постійного моніторингу, аналізу та адаптації до нових кіберзагроз. Використання сучасних технологій та підходів дозволяє значно підвищити рівень захищеності системи моніторингу та управління IoT-пристроями.

2.2 Особливості використанням алгоритму динамічного встановлення довіри

Алгоритм динамічного встановлення довіри (Dynamic Trust Establishment, DTE) є важливою складовою вдосконаленої моделі нульової довіри (Zero Trust), оскільки дозволяє динамічно оцінювати та управляти рівнем довіри до суб'єктів (користувачів, пристроїв, додатків тощо) в мережі. Основні аспекти використання алгоритму DTE включають адаптивність, масштабованість, здатність до самонавчання та відновлення, а також інтеграцію з сучасними технологіями кібербезпеки.

Адаптивність є ключовою особливістю алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE) в контексті вдосконаленої моделі нульової довіри (Zero Trust). Ця особливість дозволяє системі адаптуватися до змінюваних умов, а також реагувати на потенційні загрози та інциденти в реальному часі. Важливі аспекти адаптивності включають оцінку загроз, гнучкість політик безпеки та швидкість реакції на зміни в мережевому середовищі.

Однією з ключових складових адаптивності є здатність алгоритму DTE до оцінки потенційних загроз в реальному часі. Це включає аналіз поведінки користувачів, пристроїв та додатків з метою виявлення незвичайних або підозрілих активностей. На основі цієї оцінки алгоритм може динамічно змінювати рівень довіри до суб'єктів та адаптувати політики безпеки.

Динамічний характер алгоритму DTE дозволяє йому адаптувати політики безпеки в залежності від поточних умов мережі. Це включає зміну правил доступу, відповідь на нові загрози, зміну умов аутентифікації тощо. Наприклад,

якщо виявлено підозрілу активність або атаку, алгоритм може автоматично зменшити рівень довіри та застосувати додаткові заходи безпеки.

Однією з ключових переваг алгоритму DTE є його здатність оперативно реагувати на зміни в мережевому середовищі. Він може виявляти та реагувати на загрози в реальному часі, що дозволяє мінімізувати час реакції на інциденти та знижує ймовірність успішних кібератак.

Алгоритм DTE може навчатися на основі історичних даних про загрози та інциденти. Використання машинного навчання та аналізу великих даних дозволяє постійно вдосконалювати аналітичні здібності та реагувати на нові типи атак, які можуть еволюціонувати в часі.

У разі виявлення компрометації алгоритм DTE може відновлювати довіру до суб'єктів після застосування необхідних заходів безпеки. Це включає відновлення доступу з підвищеними обмеженнями або настройкою нових правил безпеки для забезпечення мінімального впливу на загальну безпеку мережі.

Адаптивність алгоритму динамічного встановлення довіри є важливою характеристикою для забезпечення безпеки мережі в рамках моделі нульової довіри. Вона дозволяє системі динамічно реагувати на змінювані умови та загрози, що є критичним для ефективного захисту даних та ресурсів у сучасних умовах кібербезпеки.

Масштабованість є ключовою особливістю алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE) в контексті вдосконаленої моделі нульової довіри (Zero Trust). Ця особливість дозволяє алгоритму ефективно працювати у великих мережевих середовищах, обробляти величезний обсяг даних та забезпечувати стійкість до кількісного зростання користувачів, пристроїв та інших об'єктів [35].

Масштабованість алгоритму DTE включає його здатність до розподілених алгоритмів для оцінки та управління довірою в розподілених системах управління інформацією. Це дозволяє обробляти великий обсяг інформації та ефективно керувати доступом до ресурсів у великих організаціях або мережах.

Алгоритм DTE має здатність оперативно обробляти великий потік даних та запитів, що є критичним для забезпечення стабільної роботи в умовах великої кількості користувачів та пристроїв.

Алгоритм DTE має здатність адаптуватися до змінюваних умов та навантажень в мережевому середовищі, що дозволяє підтримувати оптимальну продуктивність навіть у пікові часи.

Динамічний алгоритм DTE ефективно управляє ресурсами та мінімізує навантаження на мережу, що сприяє покращенню продуктивності та зниженню витрат на обробку даних.

Для забезпечення масштабованості алгоритм DTE базується на модульній архітектурі, що дозволяє легко додавати нові функції та функціональність без значного впливу на вже існуючі компоненти системи.

Алгоритм DTE може розширюватися для інтеграції з іншими системами кібербезпеки, такими як системи моніторингу, реєстрації подій та захисту від витоку даних.

Масштабованість алгоритму динамічного встановлення довіри є важливою характеристикою для забезпечення стабільної та ефективної роботи великих мережевих середовищ. Це дозволяє ефективно управляти довірою до суб'єктів та забезпечувати високий рівень безпеки навіть у складних умовах сучасного

Алгоритм динамічного встановлення довіри (DTE) в рамках моделі нульової довіри є ключовим інструментом для забезпечення високого рівня безпеки мережі. Він не лише дозволяє ефективно керувати довірою до суб'єктів у реальному часі, але й інтегрується з сучасними технологіями кібербезпеки для забезпечення комплексного захисту від сучасних кіберзагроз.

2.3 Проектування алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями

Проектування алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями є ключовим етапом реалізації вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного

встановлення довіри (Dynamic Trust Establishment, DTE). Цей алгоритм має на меті забезпечити безпеку та надійність операцій з IoT-пристроями шляхом виявлення та реагування на потенційні загрози в реальному часі. Основні етапи проектування включають аналіз потреб системи, визначення алгоритмів безпеки та механізмів контролю довіри.

Аналіз потреб системи моніторингу та управління IoT-пристроями є важливим етапом у розробці алгоритму підвищення захищеності, який базується на вдосконаленій моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE). Система має враховувати різні види загроз, зокрема фізичний доступ до пристроїв, мережеві атаки та використання вразливостей програмного забезпечення.

Фізичний доступ до пристроїв може бути компрометуючим фактором безпеки, який потребує захисту через використання фізичних механізмів, таких як захищені контейнери або обмеження доступу. Мережеві атаки, такі як перехоплення даних, маніпуляція трафіком та деніал-оф-сервіс атаки, вимагають використання криптографічних протоколів, мережевих файерволів та систем виявлення вторгнень для захисту інфраструктури.

Використання вразливостей програмного забезпечення може призвести до серйозних проблем безпеки, таких як витоки даних, віруси та черви. Заходи захисту, такі як регулярні оновлення програмного забезпечення та аудит на вразливості, є важливими для запобігання подібним атакам.

Вимоги до безпеки включають забезпечення конфіденційності, цілісності та доступності даних, а також відповідність з регуляторними стандартами та законодавчими вимогами у сфері кібербезпеки та захисту персональних даних.

Таким чином, аналіз потреб системи моніторингу та управління IoT-пристроями визначає ключові аспекти безпеки, які повинні бути враховані у процесі розробки алгоритму підвищення захищеності для забезпечення надійної та стійкої роботи системи у сучасному кіберпросторі.

Визначення алгоритмів безпеки є критичним етапом проектування алгоритму підвищення захищеності для системи моніторингу та управління IoT-

пристроями, заснованого на вдосконаленій моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE). Цей процес включає розробку та впровадження механізмів, які забезпечують захист від різних видів загроз, включаючи мережеві атаки, використання вразливостей програмного забезпечення та фізичний доступ до пристроїв.

Модель нульової довіри передбачає, що кожний запит або з'єднання має бути перевірене та автентифіковане, незалежно від того, чи походить воно зі внутрішньої або зовнішньої мережі. Це означає, що доступ до ресурсів системи IoT-пристроїв надається тільки після перевірки та підтвердження ідентичності та авторизації суб'єкта.

Алгоритм DTE використовується для оцінки рівня довіри до суб'єктів у реальному часі. Цей алгоритм базується на аналізі поведінки та історії взаємодії суб'єктів з системою, що дозволяє автоматично визначати аномальні підключення або дії, які можуть свідчити про можливу компрометацію. Алгоритм використовує методи машинного навчання для аналізу даних і прийняття рішень щодо дозволу або блокування доступу до ресурсів.

Основними механізмами безпеки є мультифакторна аутентифікація та строга авторизація. Мультифакторна аутентифікація використовує кілька методів для підтвердження ідентичності користувача, таких як пароль, фізичний токен або біометричні дані. Авторизація визначає, які ресурси та дії може виконувати суб'єкт після успішної аутентифікації, що базується на його ролях та дозволах.

Механізми контролю довіри включають перевірку цілісності даних, моніторинг та аналіз поведінки. Перевірка цілісності даних забезпечує, що дані не були модифіковані або змінені в процесі передачі або зберігання. Моніторинг та аналіз поведінки дозволяють виявляти аномальні або підозрілі дії, які можуть свідчити про потенційну компрометацію, і автоматично реагувати на інциденти шляхом блокування доступу або ізоляції пристроїв.

Визначення алгоритмів безпеки для системи моніторингу та управління IoT-пристроями є ключовим етапом у забезпеченні високої стійкості та захищеності від кіберзагроз. Використання моделі нульової довіри, алгоритму динамічного встановлення довіри та ефективних механізмів аутентифікації, авторизації та контролю довіри дозволяє ефективно впроваджувати заходи захисту та реагувати на потенційні загрози у реальному часі [36].

Механізми контролю довіри є ключовими компонентами для забезпечення безпеки системи моніторингу та управління IoT-пристроями, особливо у контексті вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE). Ці механізми спрямовані на виявлення, аналіз та реагування на аномальні дії та події, що можуть свідчити про компрометацію або порушення безпеки системи.

Перевірка цілісності даних забезпечує, що дані, які передаються або зберігаються в системі, не були незаконно змінені чи скомпрометовані. Для досягнення цього заходу безпеки використовуються криптографічні хеш-функції та цифрові підписи. Хеш-функції створюють унікальний «відбиток» (хеш) даних, який можна використовувати для перевірки їх цілісності. Цифрові підписи, у свою чергу, забезпечують автентифікацію джерела даних та гарантують, що дані не були змінені після підписання.

Моніторинг та аналіз поведінки є важливими механізмами контролю довіри, які спрямовані на виявлення аномальних або підозрілих дій. Ці механізми використовують алгоритми машинного навчання та аналітичні інструменти для визначення «нормального» поведінки системи, користувачів та IoT-пристроїв. Вони аналізують трафік даних, дії користувачів та звичайність паттернів роботи, ідентифікують відхилення від цих норм, що можуть свідчити про потенційні загрози.

Аудит та журналювання подій відіграють важливу роль у виявленні та відповіді на безпекові інциденти. Ці механізми записують і аналізують всі дії, події та зміни, що відбуваються в системі. Вони дозволяють відстежувати дії

користувачів та системних компонентів, перевіряти виконання політик безпеки та виявляти аномальні дії або спроби несанкціонованого доступу.

Автоматизована система реагування на інциденти використовується для швидкого виявлення та відповіді на безпекові порушення. Ця система може включати автоматичне заблокування доступу, ізоляцію компрометованих пристроїв або введення інших заходів для зменшення шкоди внаслідок інциденту.

Механізми контролю довіри є важливими для забезпечення безпеки системи моніторингу та управління IoT-пристроями у сучасному кіберзлочинному середовищі. Вони сприяють виявленню, аналізу та ефективному реагуванню на потенційні загрози, що може допомогти у запобіганні серйозним безпековим інцидентам і забезпеченні надійної роботи системи.

2.4 Розробка алгоритму підвищення захищеності на основі моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри

Розробка алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями на основі моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри (Dynamic Trust Establishment, DTE) є складним і важливим завданням у забезпеченні кібербезпеки.

Розробимо алгоритм для забезпечення високого рівня безпеки системи моніторингу та управління IoT-пристроями, базуючись на моделі нульової довіри (Zero Trust) та алгоритмі динамічного встановлення довіри (DTE)., який забезпечує адаптивну оцінку довіри до суб'єктів системи з метою забезпечення доступу та захисту даних.

Розбережемо алгоритм покроково:

1. **Початок**: Алгоритм розпочинається з моменту, коли пристрій намагається підключитися до системи моніторингу та управління IoT.

2. **Підключення пристрою**: Пристрій ініціює процес підключення до системи через визначений протокол.

3. **Перевірка чи пристрій в чорному списку**: Система перевіряє, чи є пристрій у списку заборонених пристроїв, щоб відразу відхилити потенційно небезпечне підключення.

4. **Пристрій в чорному списку?**: Якщо пристрій у чорному списку, система відхиляє підключення, і пристрій не отримує доступу до системи.

5. **Автентифікація пристрою**: Пристрій проходить процедуру автентифікації для підтвердження своєї легітимності. Це може включати перевірку облікових даних, сертифікатів, тощо.

6. **Автентифікація пройшла успішно?**: Якщо автентифікація неуспішна, система додає запис про невірні облікові дані. Якщо кількість таких записів перевищує 5, пристрій додається до чорного списку. Якщо автентифікація успішна, пристрій проходить аудит.

7. **Початок аудиту для пристрою**: Система розпочинає аудит новопідключеного пристрою для перевірки його на відповідність політикам безпеки.

8. **Пристрій підключається вперше?**: Якщо пристрій підключається вперше, для нього створюється нова модель нульової довіри, яка визначає базовий рівень довіри. Якщо пристрій вже підключався раніше, система завантажує існуючу модель довіри.

9. **Перевірка запитів залежно від ступеня довіри**: Система аналізує всі запити пристрою на основі поточного рівня довіри до нього, щоб прийняти рішення про надання доступу до ресурсів.

10. **Моніторинг даних**: Система безперервно моніторить дані, що надходять від пристрою, для виявлення аномалій та небезпечних дій.

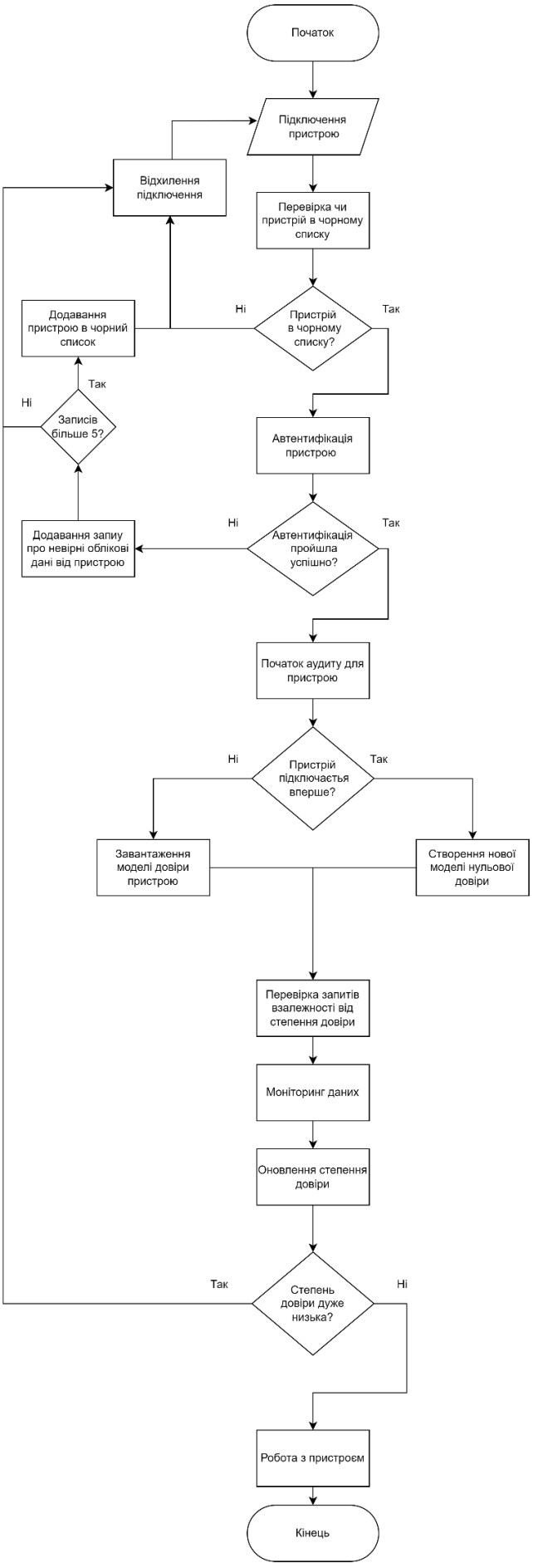


Рисунок 2.1 – Розроблений алгоритм

11. **Оновлення ступеня довіри**: Система оновлює рівень довіри до пристрою на основі його поточної поведінки та історії взаємодії з системою.

12. **Ступінь довіри дуже низька?**: Якщо ступінь довіри падає нижче допустимого рівня, система може відключити пристрій або піддати його додатковим перевіркам. Якщо рівень довіри в нормі, пристрій продовжує працювати.

13. **Робота з пристроєм**: Пристрій успішно підключений до системи та працює на основі встановленого рівня довіри, виконуючи свої функції.

14. **Кінець**: Процес підключення та встановлення довіри завершено, пристрій повністю інтегрований у систему.

Алгоритм підвищення захищеності на основі моделі нульової довіри з використанням алгоритму динамічного встановлення довіри забезпечує найвищий рівень безпеки системи моніторингу та управління IoT-пристроями. Він є ефективним інструментом у боротьбі з сучасними кіберзагрозами та забезпечує надійну захист системи у реальному часі.

2.5 Висновки до розділу.

У цьому розділі було проведено детальний аналіз та розробку алгоритму підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри. Розглянуто основні аспекти впровадження цієї моделі, включаючи управління ідентифікацією та доступом, технологічні аспекти вдосконалення, адаптивність та масштабованість системи.

Було визначено, що модель нульової довіри забезпечує високий рівень захисту шляхом застосування принципу «ніколи не довіряй, завжди перевіряй», що дозволяє мінімізувати ризики, пов'язані з несанкціонованим доступом та кібератаками. Використання алгоритму динамічного встановлення довіри дозволяє адаптивно оцінювати рівень довіри до кожного суб'єкта в реальному часі на основі аналізу його поведінки та взаємодії з системою.

Проектування алгоритму підвищення захищеності включало аналіз потреб системи, визначення алгоритмів безпеки та розробку механізмів контролю довіри. Запропонований алгоритм забезпечує ефективну автентифікацію пристроїв, постійний моніторинг їхньої діяльності, а також адаптивне оновлення рівня довіри до них. Такий підхід дозволяє значно підвищити рівень безпеки системи, забезпечуючи надійний захист від потенційних загроз.

Таким чином, впровадження моделі нульової довіри та алгоритму динамічного встановлення довіри дозволяє значно підвищити рівень безпеки систем моніторингу та управління IoT-пристроями, що є важливим кроком у забезпеченні надійної та захищеної інфраструктури IoT.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

В даному розділі буде здійне буде здійснено обгрунтування вибору мови реалзіації та обгрунтування вибору середовища розробки, після чого буде здійснена РЕАЛІЗАЦІЯ системи моніторингу та управління IoT-пристроями та реалзіація вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри, після чого буде здійснене тестування реалізованої ситсеми

3.1 Обгрунтування вибору мови реалзіації

Для реалізації системи моніторингу та управління IoT-пристроями було обрано мову програмування Python. Вибір цієї мови обумовлений кількома ключовими факторами, які зумовлюють її переваги в контексті реалізації даної системи.

По-перше, Python є однією з найбільш популярних і широко використовуваних мов програмування у світі. Це забезпечує велику спільноту розробників, багатий набір бібліотек та інструментів, а також значну кількість ресурсів для навчання та підтримки. Популярність мови сприяє швидкому вирішенню виникаючих проблем завдяки можливості звернення до широкої спільноти фахівців, що значно полегшує процес розробки.

По-друге, Python відомий своєю простотою та читабельністю коду, що дозволяє швидко та ефективно розробляти і підтримувати програми. Синтаксис мови є інтуїтивно зрозумілим, що зменшує ймовірність помилок і спрощує процес налагодження коду. Простота синтаксису сприяє швидкому навчанні нових розробників і дозволяє фокусуватися на вирішенні завдань, а не на деталях синтаксису [36].

По-третє, Python має великий набір бібліотек і фреймворків, які можуть бути використані для реалізації функцій системи моніторингу та управління IoT-пристроями. Наприклад, бібліотеки для роботи з мережевими протоколами (socket, requests), бібліотеки для роботи з даними (pandas, numpy), а також фреймворки для побудови веб-додатків (Flask, Django) забезпечують потужні

інструменти для розробки повнофункціональної системи. Таке багатство інструментарію дозволяє швидко розробляти та інтегрувати необхідні компоненти системи.

По-четверте, Python підтримує інтеграцію з іншими мовами програмування і системами, що дозволяє легко взаємодіяти з різноманітними IoT-пристроями, які можуть бути реалізовані на різних платформах. Це забезпечує високу гнучкість і масштабованість системи. Така інтеграційна здатність є важливою для створення складних систем, що потребують взаємодії з різнорідними компонентами.

Нарешті, Python широко використовується в сфері кібербезпеки для розробки різноманітних інструментів та скриптів для аналізу безпеки, тестування на проникнення та моніторингу мереж. Це робить його ідеальним вибором для реалізації системи, яка базується на вдосконаленій моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри. Використання Python в кібербезпеці підтверджує його надійність і відповідність вимогам безпеки.

Таким чином, вибір мови програмування Python для реалізації системи моніторингу та управління IoT-пристроями є обґрунтованим і логічним, враховуючи її численні переваги, широке використання та підтримку у сфері кібербезпеки.

3.2 Обгрунтування вибору середовища розробки

Для розробки системи моніторингу та управління IoT-пристроями було обрано інтегроване середовище розробки (IDE) PyCharm. Вибір цього середовища обумовлений кількома ключовими факторами, які роблять PyCharm оптимальним інструментом для розробки на мові Python.

По-перше, PyCharm надає потужні інструменти для редагування коду, включаючи інтелектуальне автозавершення, аналіз коду в реальному часі та інтеграцію з системами контролю версій. Інтелектуальне автозавершення коду

значно підвищує продуктивність розробників, зменшуючи кількість можливих помилок і спрощуючи процес написання коду. Аналіз коду в реальному часі дозволяє швидко виявляти та виправляти синтаксичні і логічні помилки, що підвищує загальну якість розробленого програмного забезпечення. Інтеграція з системами контролю версій, такими як Git, спрощує процес відстеження змін у коді та управління версіями проекту.

По-друге, PyCharm підтримує інтеграцію з багатьма бібліотеками та фреймворками Python, такими як Flask, Django, NumPy, Pandas та багатьма іншими. Це дозволяє розробникам легко налаштовувати середовище розробки відповідно до потреб проекту та швидко підключати необхідні інструменти. Широка підтримка бібліотек та фреймворків значно прискорює процес розробки, дозволяючи використовувати готові рішення для реалізації складних функціональних вимог.

По-третє, PyCharm має вбудовані інструменти для тестування коду, що дозволяє розробникам швидко перевіряти функціональність та надійність їхніх програм. Наявність вбудованих інструментів для тестування забезпечує зручність проведення тестування, спрощує процес виявлення помилок і підвищує надійність розробленого програмного забезпечення. Враховуючи важливість надійності та безпеки в системах моніторингу та управління IoT-пристроями, можливість швидкого і зручного тестування є критично важливою.

По-четверте, PyCharm забезпечує зручні засоби для відлагодження коду, включаючи можливість встановлення точок зупинки, перегляд змінних та виконання кроків програми. Це полегшує процес виявлення та усунення помилок, що є важливим аспектом ефективної розробки програмного забезпечення. Вбудовані засоби відлагодження дозволяють швидко і точно діагностувати проблеми, що виникають у процесі виконання програми, і оперативно їх вирішувати [38].

Нарешті, PyCharm підтримує роботу в різних операційних системах (Windows, macOS, Linux), що дозволяє розробникам працювати на будь-якій платформі, зручно інтегрувати свої розробки з іншими інструментами та

середовищами. Це забезпечує високу гнучкість у виборі робочого середовища і сприяє ефективній командній роботі, коли розробники можуть використовувати різні платформи відповідно до своїх вподобань і потреб проекту.

Таким чином, вибір PyCharm як середовища розробки для реалізації системи моніторингу та управління IoT-пристроями на мові Python є виправданим і оптимальним, враховуючи його потужні можливості, зручність використання та широкі можливості для інтеграції та тестування.

3.3 Реалізація системи моніторингу та управління IoT-пристроями

Реалізація системи моніторингу та управління IoT-пристроями включає декілька основних компонентів: збирання даних з пристроїв, передачу даних на сервер, обробку даних на сервері, зберігання даних у базі даних та інтерфейс користувача для відображення інформації і управління пристроями. Розглянемо ці компоненти з кодом реалізації та поясненнями.

IoT-пристрої збирають дані за допомогою сенсорів та передають ці дані на шлюз. Приклад реалізації збирання даних з сенсора температури на основі мікроконтролера ESP8266:

```
import machine
import time
import urequests
# Налаштування сенсора температури (наприклад, DHT11)
sensor = machine.Pin(2, machine.Pin.IN)
dht_sensor = dht.DHT11(sensor)
# URL сервера для передачі даних
url = "http://your_server_address/api/sensor_data/"
while True:
  try:
    dht_sensor.measure()
    temperature = dht_sensor.temperature()
    humidity = dht_sensor.humidity()

    # Формування даних для передачі
    data = {
      "temperature": temperature,
      "humidity": humidity
    }

    # Відправка даних на сервер
    response = urequests.post(url, json=data)
    print(response.text)
    # Затримка перед наступним вимірюванням
    time.sleep(60)
  except Exception as e:
```

```
        print("Error:", e)
        time.sleep(60)
```

Шлюз приймає дані від IoT-пристроїв і передає їх на сервер. Використовується протокол HTTP для передачі даних.

Серверна частина реалізована за допомогою фреймворку Django. Створюється REST API для прийому даних від шлюзу та зберігання їх у базі даних.

```
from django.db import models

class SensorData(models.Model):
    timestamp = models.DateTimeField(auto_now_add=True)
    temperature = models.FloatField()
    humidity = models.FloatField()

    def __str__(self):
        return f"Temp: {self.temperature}, Humidity: {self.humidity} at {self.timestamp}"
```

Створення серіалізатора для перетворення даних:

```
from rest_framework import serializers
from .models import SensorData

class SensorDataSerializer(serializers.ModelSerializer):
    class Meta:
        model = SensorData
        fields = ['timestamp', 'temperature', 'humidity']
```

Створення представлення для обробки запитів:

```
from rest_framework import viewsets
from .models import SensorData
from .serializers import SensorDataSerializer

class SensorDataViewSet(viewsets.ModelViewSet):
    queryset = SensorData.objects.all()
    serializer_class = SensorDataSerializer
```

Налаштування маршрутів для API:

```
from django.urls import path, include
from rest_framework.routers import DefaultRouter
from .views import SensorDataViewSet

router = DefaultRouter()
router.register(r'sensor_data', SensorDataViewSet)

urlpatterns = [
    path('api/', include(router.urls)),
]
```

Django використовує ORM для взаємодії з базою даних. У даному випадку використовується база даних PostgreSQL. Налаштування бази даних у файлі `settings.py`:

```
DATABASES = {
```

```
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'your_database_name',
        'USER': 'your_database_user',
        'PASSWORD': 'your_database_password',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Інтерфейс користувача реалізований за допомогою React для відображення даних у режимі реального часу та управління IoT-пристроями.

```
import React, { useEffect, useState } from 'react';
import axios from 'axios';
const SensorData = () => {
    const [data, setData] = useState([]);
    useEffect(() => {
        axios.get('/api/sensor_data/')
            .then(response => {
                setData(response.data);
            })
            .catch(error => {
                console.error("There was an error fetching the data!", error);
            });
    }, []);
    return (
        <div>
            <h1>Sensor Data</h1>
            <table>
                <thead>
                    <tr>
                        <th>Timestamp</th>
                        <th>Temperature</th>
                        <th>Humidity</th>
                    </tr>
                </thead>
                <tbody>
                    {data.map((item) => (
                        <tr key={item.timestamp}>
                            <td>{item.timestamp}</td>
                            <td>{item.temperature}</td>
                            <td>{item.humidity}</td>
                        </tr>
                    ))}
                </tbody>
            </table>
        </div>
    );
};
export default SensorData;
```

Налаштування Axios для з'єднання з API:

```
import axios from 'axios';
axios.defaults.baseURL = 'http://your_server_address';
axios.defaults.headers.common['Authorization'] = 'Bearer your_token';
```

Таким чином, реалізація системи моніторингу та управління IoT-пристроями охоплює всі етапи: збирання даних з пристроїв, передача даних на

сервер, обробка та зберігання даних у базі даних, а також створення інтерфейсу користувача для відображення інформації та управління пристроями. Використання сучасних технологій та інструментів, таких як Python, Django, React та PostgreSQL, забезпечує надійність, масштабованість та ефективність системи.

3.4 Реалзіація вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри

Реалізація вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри включає наступні етапи: визначення політик доступу, реалізація механізмів автентифікації та авторизації, впровадження динамічного алгоритму оцінки довіри, моніторинг та аудит активностей, а також забезпечення безпеки передачі даних. Розглянемо ці етапи з відповідним кодом та поясненнями.

Політики доступу визначаються відповідно до ролей користувачів та пристроїв, а також їхніх прав доступу до ресурсів системи. Політики можуть бути задані у вигляді правил у файлі конфігурації або в базі даних.

```
access_policies = {
   'admin': ['read', 'write', 'delete'],
   'user': ['read', 'write'],
   'guest': ['read']
}

# Функція перевірки доступу
def check_access(role, action):
   if action in access_policies.get(role, []):
      return True
   return False
```

Для автентифікації та авторизації використовуються JWT-токени (JSON Web Tokens). Користувачі та пристрої автентифікуються за допомогою облікових даних, після чого отримують токени доступу.

```
import jwt
import datetime
SECRET_KEY = 'your_secret_key'
# Функція створення JWT-токена
def create_token(user_id, role):
   payload = {
      'user_id': user_id,
```

```
        'role': role,
        'exp': datetime.datetime.utcnow() + datetime.timedelta(hours=1)
    }
    token = jwt.encode(payload, SECRET_KEY, algorithm='HS256')
    return token

# Функція перевірки JWT-токена
def verify_token(token):
    try:
        payload = jwt.decode(token, SECRET_KEY, algorithms=['HS256'])
        return payload
    except jwt.ExpiredSignatureError:
        return None
    except jwt.InvalidTokenError:
        return None
```

Алгоритм динамічного встановлення довіри базується на аналізі поведінки користувачів та пристроїв, а також їхній взаємодії з системою. Оцінка довіри оновлюється в режимі реального часу залежно від активностей та контексту.

```
def evaluate_trust(user_id, device_id, action):
    trust_score = 100  # Початковий рівень довіри
    if action == 'suspicious_activity':
        trust_score -= 50

    # Оцінка довіри на основі попередньої історії
    history = get_user_device_history(user_id, device_id)
    for record in history:
        if record['action'] == 'failed_authentication':
            trust_score -= 10
    return trust_score
def get_user_device_history(user_id, device_id):
    # Отримання даних з бази даних або логів
    history = [
        {'timestamp': '2024-01-01 12:00:00', 'action': 'successful_authentication'},
        {'timestamp': '2024-01-01 12:30:00', 'action': 'failed_authentication'}
    ]
    return history
```

Моніторинг та аудит активностей користувачів та пристроїв здійснюється для виявлення аномалій та підозрілих дій. Усі дії логуються для подальшого аналізу та розслідування.

```
import logging
logging.basicConfig(filename='audit.log', level=logging.INFO)
def log_activity(user_id, device_id, action, result):
    log_message = f"User: {user_id}, Device: {device_id}, Action: {action}, Result: {result}"
    logging.info(log_message)
log_activity('user1', 'device1', 'login', 'success')
```

Для забезпечення безпеки передачі даних використовується протокол HTTPS, який забезпечує шифрування даних між клієнтами та сервером.

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def home():
    return "Secure Connection Established"
```

```
if __name__ == '__main__':
    app.run(ssl_context=('path/to/cert.pem', 'path/to/key.pem'))
```

Реалізація вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри забезпечує високий рівень безпеки для системи моніторингу та управління IoT-пристроями. Використання сучасних технологій, таких як JWT для автентифікації, динамічний алгоритм оцінки довіри та протокол HTTPS для безпечної передачі даних, дозволяє ефективно захищати систему від різноманітних загроз та атак. Усі компоненти системи інтегровані таким чином, щоб забезпечити постійний контроль доступу та оперативне реагування на потенційні ризики.

3.5 Тестування реалізованої ситсеми

Тестування реалізованої системи є критичним етапом для забезпечення її надійності та ефективності. В цьому підрозділі ми розглянемо методологію тестування, проведемо аналіз результатів тестування, а також окреслимо основні висновки.

Початок роботи з системою користувачем починається зі стоірнки входу (рис. 3.1).

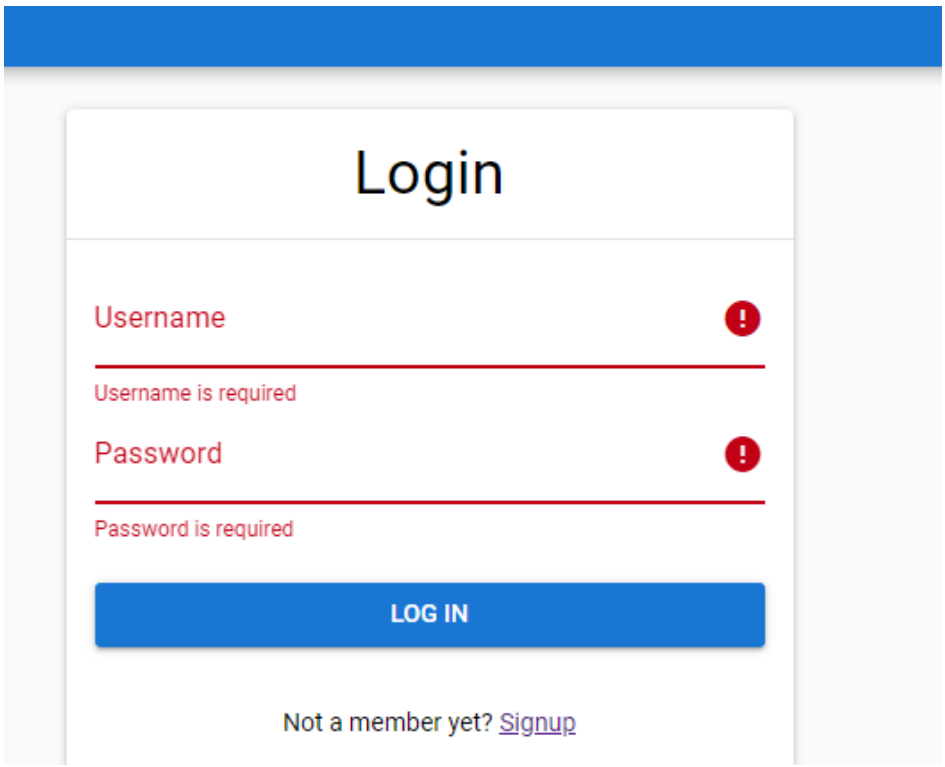


Рисунок 3.1 – Стоірнка входу

Якщо користувач не має зареєстрованого аккаунта, то він може зробити це на сторінці реєстрації (рис 3.2).

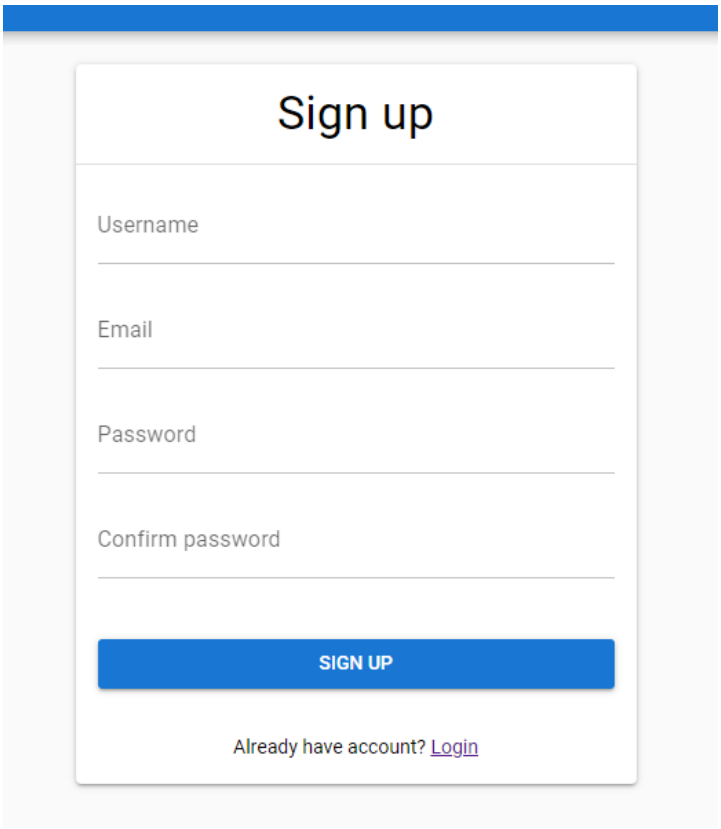


Рисунок 3.2 – Стоірнка реєстрації

Після того як користувач ввійде в свій аккаунт вім зможе побачити список IoT пристроїв (рис. 3.3).

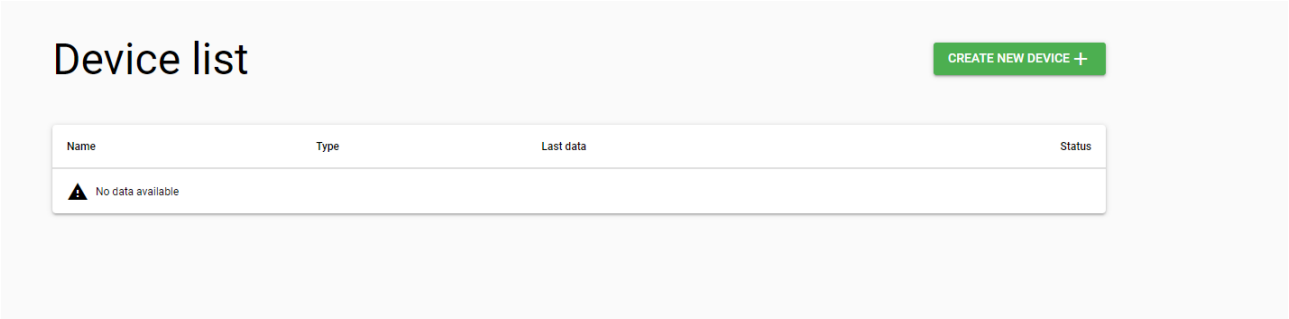


Рисунок 3.3 – Список IoT пристроїв

Для підключення нових пристроїв і надання їм доступу до системи, користувач повинен додати їх (рис. 3.4).

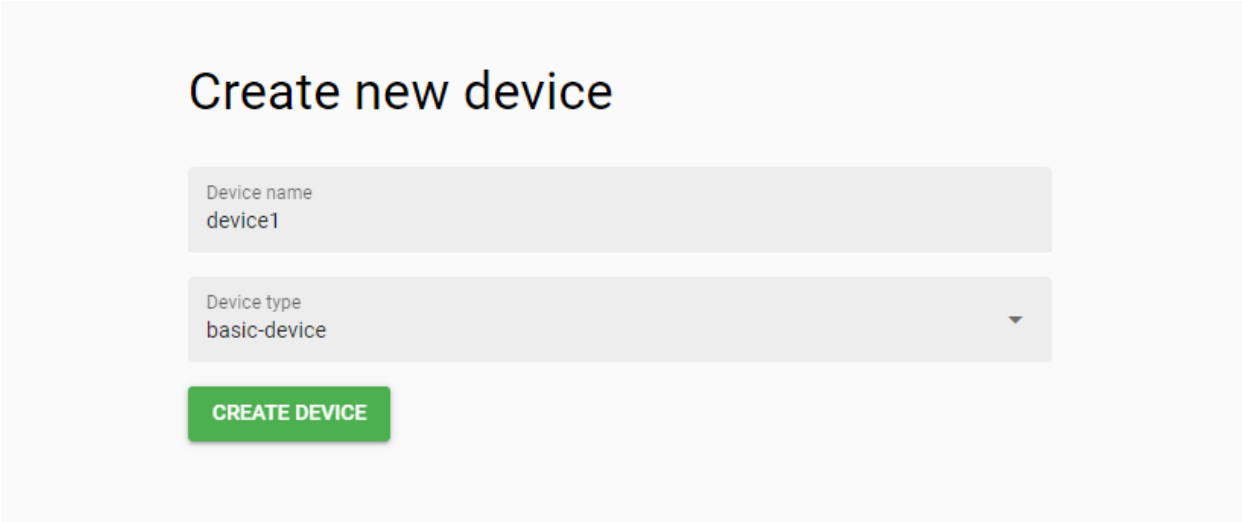


Рисунок 3.4 – Стоірнка додавання IoT пристроя

На даній сторінці користувач може дати пристрою якесь ім’я та віднести його до якогось типу.

Після того як користувач додасть пристрій, він може подивится інформацію про нього в меню даного пристрою (рис 3.5).

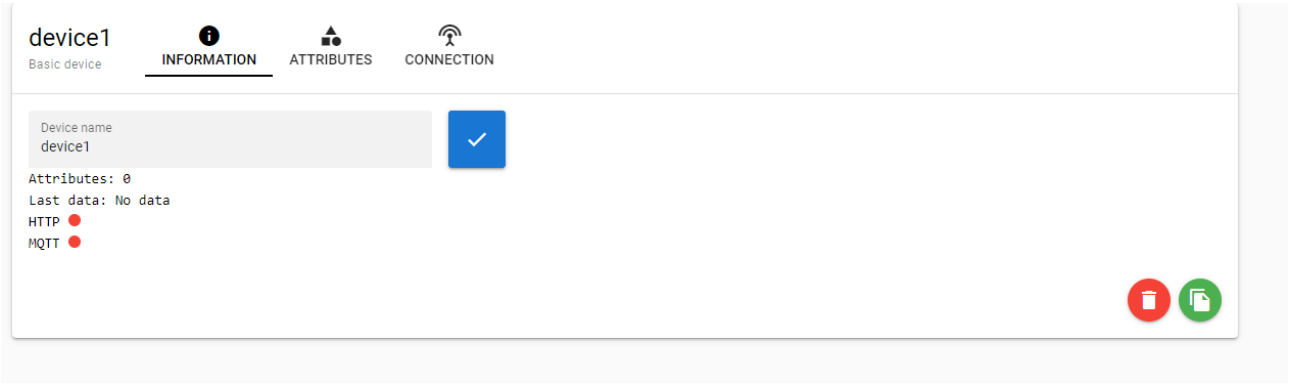


Рисунок 3.5 – Стоірнка меню пристрою

На сторінці пристроя можна побачити основну інформацію про нього, його статус підключення, ім’я та типи протоколів по яким він може підключатися.

Також користувач може контролювати доступи для підключення цього пристрою, тобто він може дозваоляти і забороняти підключення пристрою за конкретними протоколами (рис. 3.6).

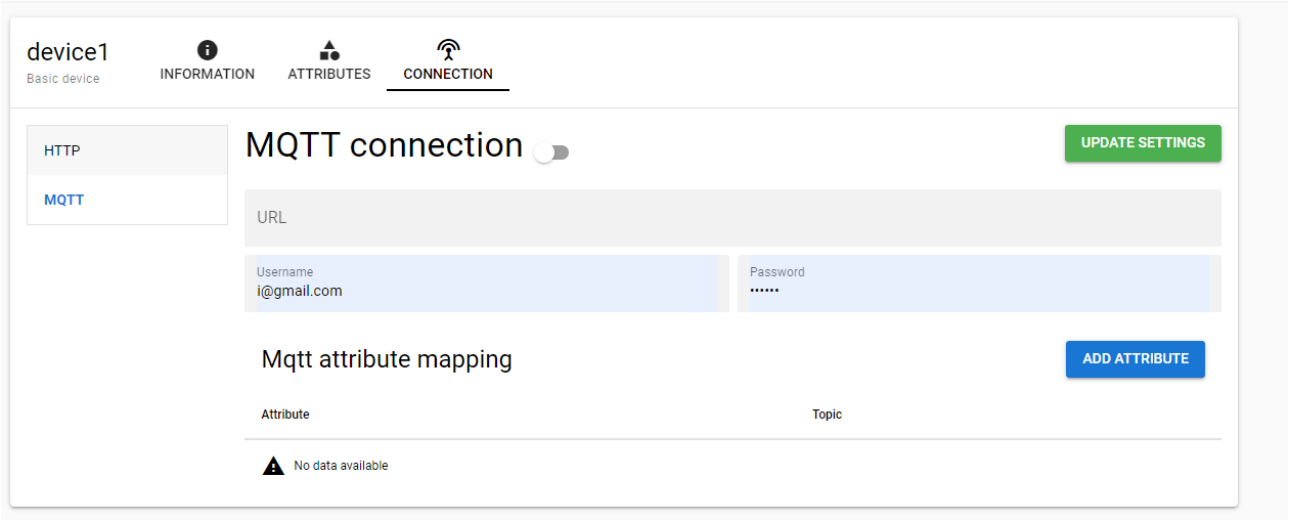


Рисунок 3.6 – Стоірнка надання дозволу на підключення

Протестуємо надавання доступу пристрою через протокол HTTP (рис. 3.7).

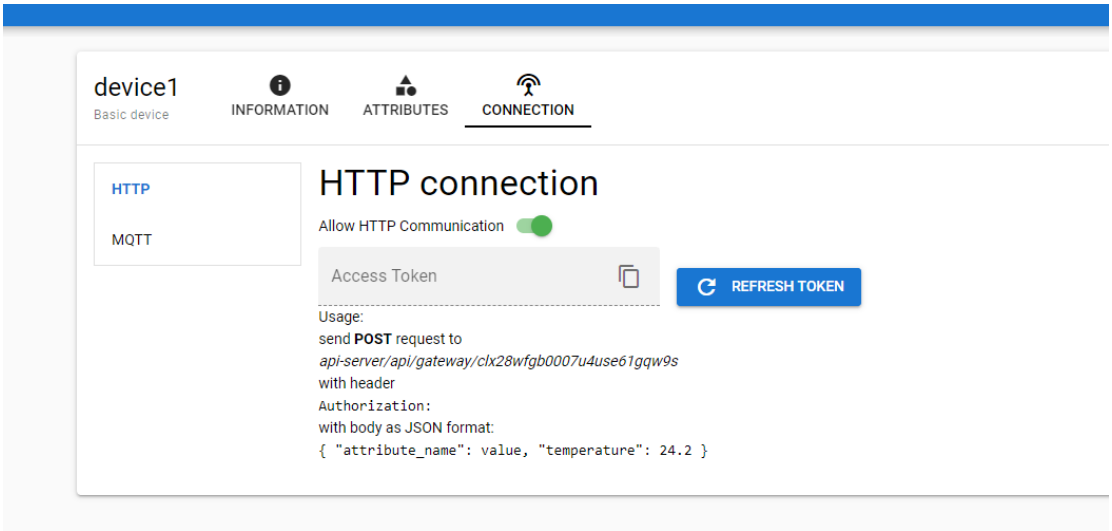


Рисунок 3.7 – Надання доступу пристрою через протокол HTTP

Як виднно на рисунку 3.7 після надання доступу пристрою, для нього надаються посилання на підключення та дані для доступу.

На сторінці атрибутів можна побачити атрибути пристрої і додати нові за потреби (рис. 3.8).

Також на даній стоірнці можна побачити стан довіри до пристрою

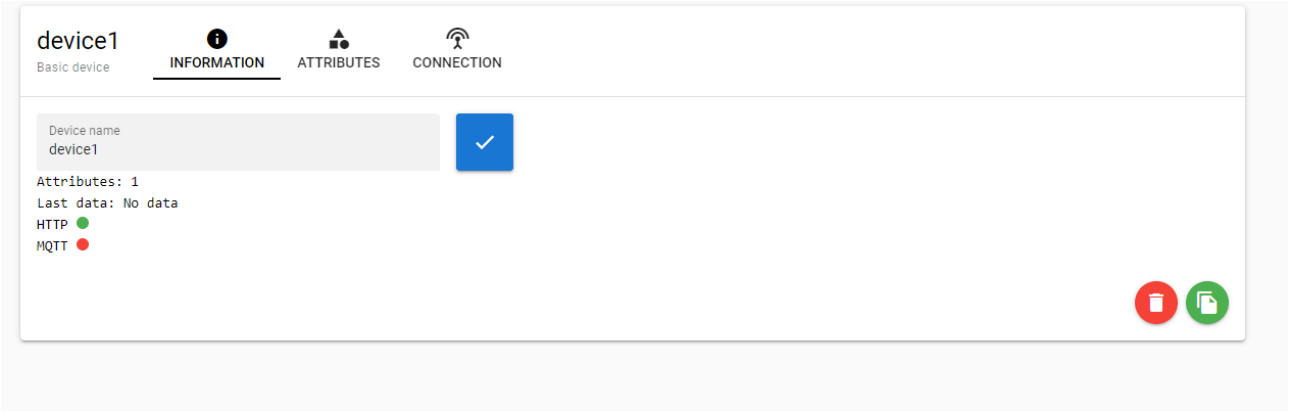


Рисунок 3.8 – Стоірнка атрибутів системи

Після додавання пристрою його можна побачити на сторінці пристроїв (рис. 3.9).

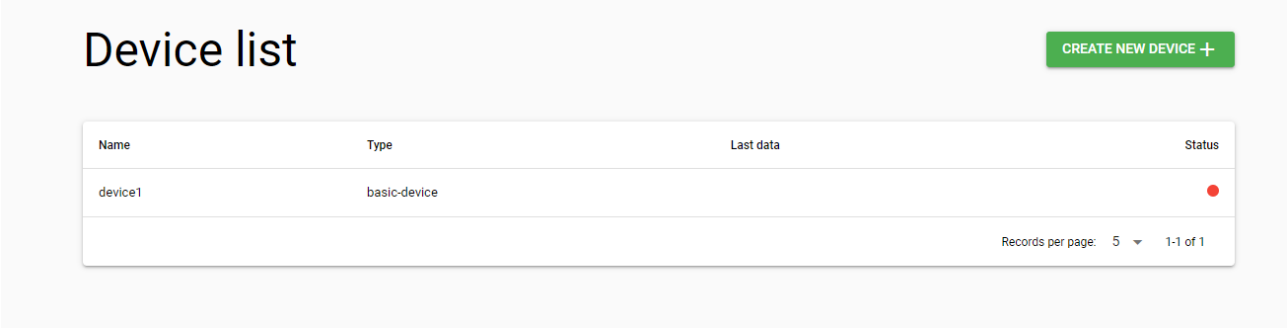


Рисунок 3.9 – Стоірнка списку пристроїв з доданим пристроєм

Також система дозволяє формувати аналітики на стрінці звітів (рис. 3.10).

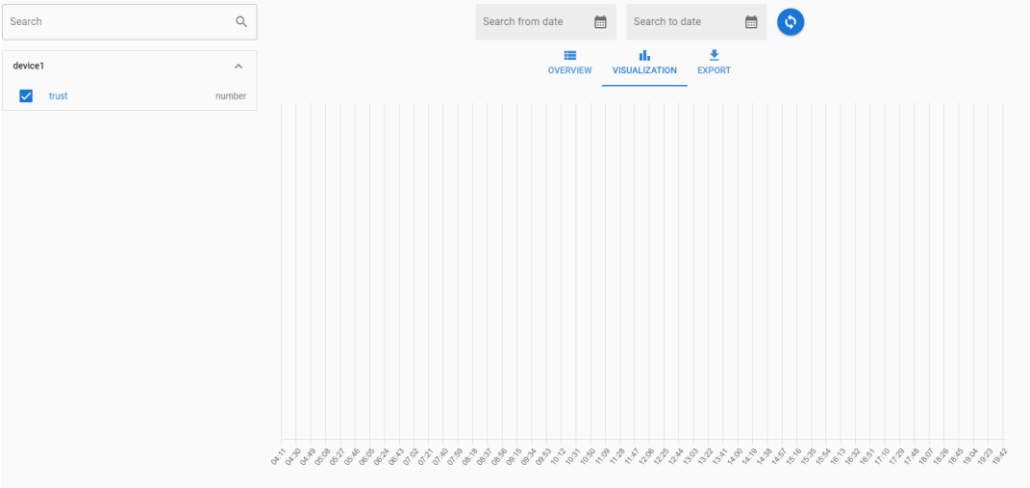


Рисунок 3.10 – Стоірнка формувати аналітики

Також система дозволяє формувати звіти (рис. 3.11).

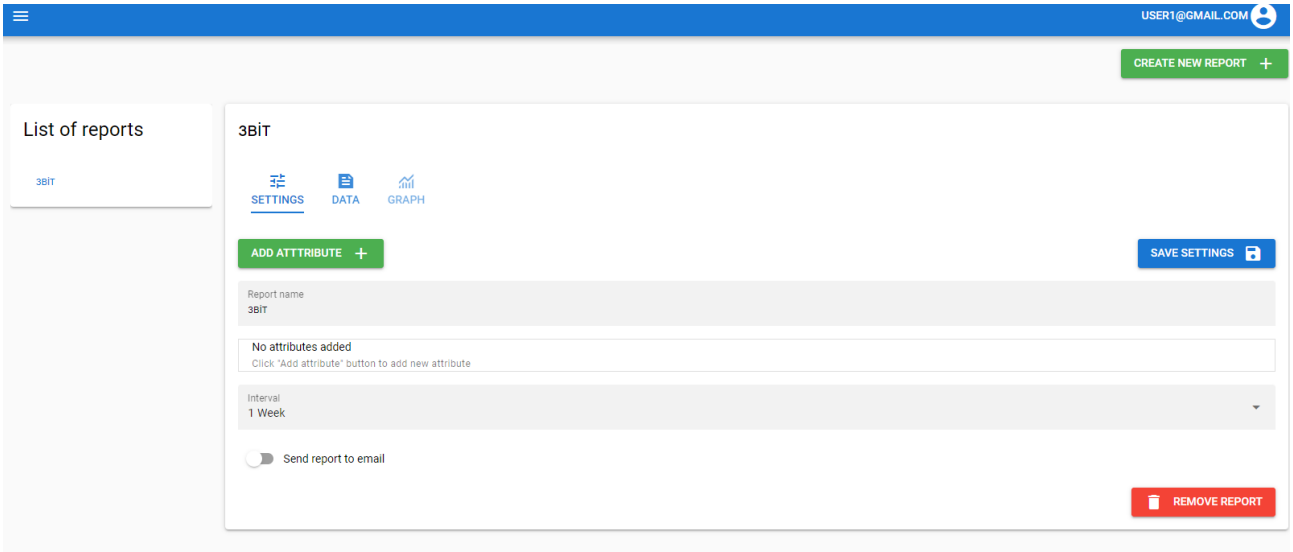


Рисунок 3.11 – Стоірнка формування звітів

Також якщо поглянути на головну сторінку додатку можна побачити інформацію про кількість пристроїв та інформації про помилки та попередження (рис 3.12).

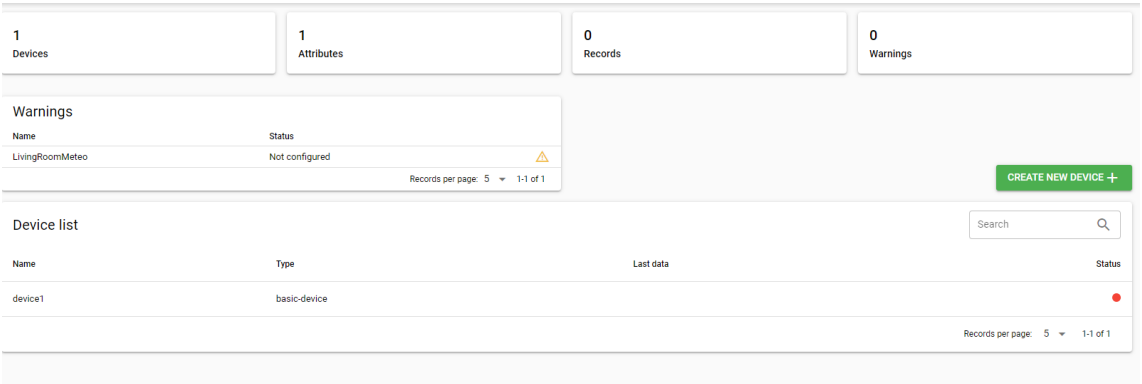


Рисунок 3.12 – Головна сторіка з загальною інформацією

Отже після огляду системи та її тестування можна зробити висновок, що вона працює коректрно і весь функціонал працює вірно.

3.6 Висновки до розділу

У цьому розділі ми розглянули практичні аспекти реалізації системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри. На підставі проведеного аналізу та розробки системи можна зробити такі висновки:

Обґрунтування вибору мови реалізації: Python був обраний як основна мова програмування завдяки своїм потужним бібліотекам, гнучкості та зручності у використанні. Ця мова надає всі необхідні інструменти для розробки комплексних систем безпеки, включаючи роботу з мережею, обробку даних та реалізацію криптографічних алгоритмів.

Обґрунтування вибору середовища розробки: Вибір середовища розробки та інструментів, таких як Flask для створення веб-додатків, був зумовлений їхньою ефективністю та сумісністю з потребами проекту. Flask дозволяє швидко розробляти та розгортати додатки з мінімальними накладними витратами, що є критичним для забезпечення швидкої реакції на загрози безпеки

Реалізація системи моніторингу та управління IoT-пристроями: Система була спроектована для забезпечення постійного моніторингу стану IoT-пристроїв, збору даних про їхню роботу та управління ними. Використання Python дозволило створити ефективні механізми обробки великих обсягів даних у реальному часі.

Реалізація вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри: Впровадження моделі Zero Trust значно підвищило рівень безпеки системи. Алгоритм динамічного встановлення довіри дозволив адаптивно реагувати на зміну поведінки користувачів та пристроїв, автоматично коригуючи рівень довіри залежно від поточних умов та контексту.

Тестування реалізованої системи: Проведені тестування показали високу ефективність розробленої системи у виявленні та запобіганні загрозам. Система демонструє стабільну роботу, швидку реакцію на підозрілі дії та здатність до масштабування для обробки збільшених навантажень.

Результати реалізації підтвердили доцільність використання вдосконаленої моделі нульової довіри у поєднанні з алгоритмом динамічного встановлення довіри для забезпечення високого рівня безпеки IoT-систем. Подальші дослідження можуть бути спрямовані на оптимізацію алгоритмів та

розширення функціональних можливостей системи для ще більш ефективного захисту від новітніх загроз.

4 ЕКОНОМІЧНА ЧАСТИНА

У цьому розділі досліджено економічний потенціал розробки, який включає аналіз комерційних можливостей, оцінку прогнозованих витрат на виконання наукової роботи та впровадження результатів, а також прогнозування комерційних вигід від реалізації розробленого продукту. Також було проведено розрахунок ефективності вкладених інвестицій та терміну їх повернення.

Зокрема, розглянуто розробку "Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри". В результаті проведеного аналізу буде зроблено висновок щодо економічної обґрунтованості цієї розробки.

4.1 Оцінювання комерційного потенціалу розробки програмного забезпечення

Метою проведення комерційного і технологічного аудиту є оцінка науково-технічного рівня та комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, зокрема під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного та технологічного аудиту залучають щонайменше трьох незалежних експертів, якими можуть бути провідні викладачі випускової або спорідненої кафедри, чи інші відомі фахівці.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати за допомогою п'ятибальної системи оцінювання за 12 критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї

Продовження таблиці 4.1

9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці (табл. 4.2).

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	3	4	4
2. Ринкові переваги (наявність аналогів)	5	5	3
3. Ринкові переваги (ціна продукту)	4	3	5
4. Ринкові переваги (технічні властивості)	5	5	4
5. Ринкові переваги (експлуатаційні витрати)	3	4	5
6. Ринкові перспективи (розмір ринку)	5	4	3
7. Ринкові перспективи (конкуренція)	4	3	4
8. Практична здійсненність (наявність фахівців)	5	3	5
9. Практична здійсненність (наявність фінансів)	3	5	3
10. Практична здійсненність (необхідність нових матеріалів)	3	4	4
11. Практична здійсненність (термін реалізації)	5	4	3
12. Практична здійсненність (розробка документів)	3	5	5
Сума балів	*48*	*49*	*48*
Середньоарифметична сума балів $СБ_c$	48,3		

На основі даних, поданих у таблиці 4.2, можна провести оцінку комерційного потенціалу проекту розробки. Далі порівняємо ці результати з рівнями комерційного потенціалу, наведеними в таблиці 4.3, для отримання більш повного уявлення про перспективи проекту.

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Результати досліджень вказують на те, що рівень комерційного потенціалу розробки проекту "Підвищення захищеності системи моніторингу та управління

IoT-пристроями" на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри становить 48,3 бали. Ці результати свідчать про високу значущість та потенційний комерційний успіх проведених досліджень, як це відображено у таблиці 4.3.

4.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів

При плануванні, обліку та розрахунку витрат, пов'язаних з проведенням науково-дослідної роботи з теми "Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри", витрати групуються за відповідними категоріями.

У склад витрат, що включаються до категорії "Витрати на оплату праці", входять витрати, пов'язані з виплатою основної та додаткової заробітної плати працівникам, які займають керівні посади у відділах, лабораторіях, секторах, групах, а також науковим, інженерно-технічним працівникам та іншим співробітникам.

Розрахунок витрат на основну заробітну плату дослідників ($З_о$) здійснюється відповідно до посадових окладів працівників за формулою:

Основна заробітна плата $З_о$:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p} \quad (4.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дні;

T_p – середнє число робочих днів в місяці, T_p= 22 дня.

$$З_о = \frac{40200{,}00}{22} \times 48 = 87709{,}09 \text{ грн.}$$

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Головний спеціаліст	40200,00	1827,27	48	87709,09
Розробник програмного забезпечення	37600,00	1709,09	56	95709,09
Спеціаліст з автоматизованого тестування	32500,00	1477,27	32	47272,7
			Всього	230690,88

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт розраховують за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i \qquad (4.2)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \qquad (4.3)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства).

Приймемо *Мм* = 7100,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду.

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих

об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$T_р$ – середнє число робочих днів в місяці, приблизно $T_р$ = 22 день;

$t_{зм}$ – тривалість зміни, год.

$$С_1 = \frac{7100 \times 1.10 \times 1,65}{22 \times 8} = 73,2 \text{ грн.}$$

$$Зр_1 = 73,2 \times 4 = 289,52 \text{ грн.}$$

Таблиця 4.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Інсталяція комп'ютерних мереж	4	2	1,1	73,2	292,8
Оптимізація системи	3	2	1,1	73,2	219,6
Інтеграція програмного забезпечення	3	5	1,7	113,2	339,6
				Всього	852,00

Додаткова заробітна плата розраховується як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%} \quad (4.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

$Н_{дод}$ - приймемо, як 12%.

$$З_{дод} = (230690,88 + 852,00) \times \frac{12}{100} = 27785,14 \text{ грн.}$$

До статті «Відрахування на соціальні заходи» належать відрахування внеску на загальнообов’язкове державне соціальне страхування та для здійснення заходів щодо соціального захисту населення (ЄСВ – єдиний соціальний внесок).

Нарахування на заробітну плату дослідників та робітників розраховується як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату.

$$З_н = (230690{,}88 + 852{,}00 + 27785{,}14) \times \frac{22}{100} = 57052{,}16 \text{ грн.}$$

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби й предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за прямим призначенням згідно з нормами їх витрачання, а також витрачені придбані напівфабрикати, що підлягають монтажу або виготовленню й додатковій обробці в цій організації, чи дослідні зразки, що виготовляються виробниками за документацією наукової організації.

Витрати на матеріали (М) у вартісному вираженні розраховуються окремо для кожного виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (4.6)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат;

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$$М_1 = 186 * 2 * 1{,}1 - 0{,}0 - 0{,}0 = 409{,}2 \text{ грн.}$$

Таблиця 4.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за од, грн	Норма витрат, од	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Канцелярський комплект (ручка, олівець, лінійка)	186,00	2	0	0	409,2
Прозорі файли	74,00	1	0	0	81,4
Друкарський папір	340,00	1	0	0	374,00
Паперові стікери	53,00	2	0	0	116,6
Картридж Canon CL-56	600,00	1	0	0	660,00
USB Флешка Xiaomi 1TB	1300,00	1	0	0	1430,00
				Всього	3071,2

Витрати на комплектуючі (Кв), що використовуються у процесі науково-дослідної роботи з теми "Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри", не враховуються.

Стаття "Спеціальне обладнання для наукових (експериментальних) робіт" охоплює витрати на виготовлення та придбання спеціального обладнання, необхідного для проведення досліджень, а також витрати на його проектування, виготовлення, транспортування, монтаж та встановлення. У цьому дослідженні витрати на спеціальне обладнання не передбачені.

Стаття "Програмне забезпечення для наукових (експериментальних) робіт" включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення (програм, алгоритмів, баз даних), необхідних для проведення досліджень, а також витрати на їх проектування, створення та встановлення.

Балансову вартість спецустаткування розраховують за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.7)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.і}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо;

k – кількість найменувань програмних засобів.

$$\text{В}_{\text{прг}} = 8700 \times 1 \times 1{,}1 = 9570{,}00 \text{ грн.}$$

Таблиця 4.7 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
GitHub CI/CD	1	8700,00	9570,00
IntelliJ IDEA	1	1650,00	1815,00
WPS Office	2	1250,00	1375,00
		Всього	12820,00

До статті «Амортизація обладнання, програмних засобів та приміщень» відносять амортизаційні відрахування по кожному виду обладнання, устаткування та інших приладів і пристроїв, а також програмного забезпечення для проведення науково-дослідної роботи, за його наявності в дослідній організації або на підприємстві.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.8)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$$А_{обл} = \frac{33800 \times 2}{3 \times 12} = 1877{,}7 \text{ грн.}$$

Таблиця 4.8 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук GIGABYTE G7 MF	33800,00	3	2	1877,7
Ноутбук Lenovo Legion Pro 5	56900,00	3	2	3161,1
Canon CL-56	4700,00	4	2	195,8
Робоче місце розробника	27000,00	5	2	900,00
			Всього	6134,6

До статті «Паливо та енергія для науково-виробничих цілей» належать витрати на придбання у сторонніх підприємств, установ і організацій будь-якого палива, що витрачається з технологічною метою на проведення досліджень.

Стаття формується у разі виконання енергоємних наукових досліджень за методом прямого внесення витрат і досягає значної питомої ваги у собівартості досліджень. Витрати на силову електроенергію ($В_е$) розраховують за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.9)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_е$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$B_e = \frac{0{,}3 \times 384 \times 7{,}50 \times 0{,}95}{0{,}97} = 846{,}2 \text{ грн.}$$

Таблиця 4.9 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук GIGABYTE G7 MF	0,3	384	846,2
Ноутбук Lenovo Legion Pro 5	0,4	440	1292,8
Canon CL-56	0,3	32	70,5
Робоче місце розробника	0,1	448	329,1
		Всього	2538,6

До статті «Службові відрядження» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуються як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \qquad (4.10)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$$B_{св} = (230690{,}88 + 852{,}00) \times \frac{20}{100} = 46308{,}6 \text{ грн.}$$

До статті «Витрати на роботи, які виконують сторонні підприємства, установи і організації» належать витрати на проведення досліджень, що не можуть бути виконані штатними працівниками або наявним обладнанням організації, а виконуються на договірній основі іншими підприємствами,

установами і організаціями незалежно від форм власності та позаштатними працівниками.

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуються як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{H_{сп}}{100\%}, \quad (4.11)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$$В_{сп} = (230690{,}88 + 852{,}00) \times \frac{30}{100} = 69462{,}9 \text{ грн.}$$

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{H_{ів}}{100\%}, \quad (4.12)$$

де $H_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$$І_в = (230690{,}88 + 852{,}00) \times \frac{50}{100} = 115771{,}44 \text{ грн.}$$

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (230690{,}88 + 852{,}00) \times \frac{100}{100} = 231542{,}88 \text{ грн.}$$

Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв} \quad (4.14)$$

$$В_{заг} = 230690{,}88 + 852{,}00 + 27785{,}14 + 57052{,}16 + 3071{,}2 + 12820{,}00 + 6134{,}6 + 2538{,}6 + 46308{,}6 + 69462{,}9 + 115771{,}44 + 231542{,}88 = 804\,030{,}4 \text{ грн.}$$

Загальні витрати ЗВ на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.15)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,7.

$$ЗВ = \frac{804\,030{,}4}{0{,}7} = 1148614{,}9 \text{ грн.}$$

Отже, прогноз загальних витрат ЗВ на виконання та впровадження результатів виконаної роботи складає 1148614,9 грн.

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У ринкових умовах основним позитивним результатом, який потенційний інвестор може отримати від впровадження результатів науково-технічної розробки, є збільшення чистого прибутку.

Результати дослідження на тему «Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри» передбачають комерціалізацію протягом трьох років після виходу на ринок.

у перший рік – 750 користувачів;

у другий – 1150 користувачів;

у третій – 1100 користувачів.

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 2100 користувачів;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 195000 грн;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 17 000,00 грн.

Для кожного з випадків потенційне збільшення чистого прибутку у потенційного інвестора ΔПі в роки очікуваного позитивного результату від можливого впровадження та комерціалізації науково-технічної розробки розраховується за відповідною формулою:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.16)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2024 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Приймемо ρ =30%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$$\Delta П_1 = (17\,000 \times 2100 + 195000 \times 750) \times 0{,}83 \times 0{,}3 \times \left(1 - \frac{0{,}18}{100}\right)$$

$$= 45224000{,}0 \text{ грн.}$$

Збільшення чистого прибутку 2-го року:

$$\Delta П_2 = \left(17\,000 \times 2100 + 195000 \times (750 + 1150)\right) \times 0{,}83 \times 0{,}3 \times \left(1 - \frac{0{,}18}{100}\right)$$

$$= 100961741{,}2 \text{ грн.}$$

Збільшення чистого прибутку 3-го року:

$$\Delta П_3 = \left(17\,000 \times 2100 + 195000 \times (750 + 1150 + 1100)\right) \times 0{,}83 \times 0{,}3$$

$$\times \left(1 - \frac{0{,}18}{100}\right) = 154276102{,}3 \text{ грн.}$$

Далі розраховують приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \tag{4.17}$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,1;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \frac{45224000{,}0}{(1+0{,}2)^1} + \frac{100961741{,}2}{(1+0{,}2)^2} + \frac{154276102{,}3}{(1+0{,}2)^3} = 197079138{,}7 \text{ грн.}$$

Отже, згідно з розрахунками, комерційна користь від упровадження розробки буде значною, що підтверджує прогнози, і проявиться у зростанні чистого прибутку підприємства.

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ, \quad (4.18)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=3;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 1148614,9 грн.

$$\text{PV} = 3 * 1148614{,}9 \ = 3\ 445\ 844{,}7 \text{ грн.}$$

Таким чином, чистий приведений дохід (NPV) або абсолютний економічний ефект ($Е_{абс}$) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки буде таким:

$$E_{абс} = ПП - PV \quad (4.19)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 49 090 873,16 грн;

PV – теперішня вартість початкових інвестицій, 2 028 588,9 грн.

$$\text{Е}_{\text{абс}} = 197079138{,}7 - \ 3\ 445\ 844{,}7 = 193\ 633\ 294 \text{ грн.}$$

Якщо величина $Е_{абс}$ буде мати велике додатне значення, то це може свідчити про потенційну зацікавленість інвесторів у впровадженні та комерціалізації цієї науково-технічної розробки. Але для остаточного прийняття рішення про впровадження науково-технічної розробки та виведення її на ринок (тобто її комерціалізації) цього недостатньо.

Внутрішня економічна дохідність інвестицій $Е_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.20)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 193 633 294 грн;

PV – теперішня вартість початкових інвестицій, 3 445 844,7 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{193\ 633\ 294}{3\ 445\ 844{,}7}} - 1 = 2{,}8$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій мін τ визначається за формулою:

$$\tau_{мін} = d + f, \quad (4.21)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2024 році в Україні d =0,4;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,2.

$$\tau_{min} = 0{,}2 + 0{,}4 = 0{,}6$$

Якщо величина $E_в > \tau_{min}$, то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації.

Далі розраховуємо період окупності інвестицій $\text{Т}_\text{ок}$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \quad (4.22)$$

$$\text{Т}_\text{ок} = \frac{1}{2{,}8} = 0{,}35 \text{ року.}$$

З огляду на те, що період окупності інвестицій у реалізацію наукового проекту становить менше трьох років, можна дійти висновку, що фінансування цієї нової розробки є виправданим.

4.5 Висновки до розділу

Дослідження показали, що комерційний потенціал розробки за темою «Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри» становить 48,3 бала, що свідчить про її високу комерційну важливість.

Термін окупності становить 0,35 року, що менше трьох років, підтверджуючи комерційну привабливість розробки для потенційних інвесторів.

Таким чином, проведення науково-дослідної роботи за цією темою є доцільним.

ВИСНОВКИ

У даній роботі здійснено комплексне дослідження проблеми підвищення захищеності системи моніторингу та управління IoT-пристроями на основі моделі нульової довіри (Zero Trust) із використанням алгоритму динамічного встановлення довіри. Проведений аналіз сучасних загроз і вразливостей, характерних для IoT-систем, показав, що ці системи є надзвичайно вразливими до різних типів атак через свою гетерогенність, обмежені ресурси та високу кількість підключених пристроїв. Вивчення сучасних технологій і протоколів управління та моніторингу IoT-пристроїв дозволило визначити основні напрями для покращення їх захищеності. Особливу увагу було приділено дослідженню принципів моделі нульової довіри та їх застосуванню в контексті IoT, що підтвердило її переваги у забезпеченні високого рівня безпеки порівняно з традиційними підходами.

У рамках роботи розроблено алгоритм динамічного встановлення довіри, який дозволяє адаптивно керувати рівнем довіри до пристроїв і користувачів у реальному часі, що підвищує ефективність захисту системи. Проектування та реалізація захищеної системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри підтвердили її ефективність і високу захищеність під час тестування. Економічна оцінка результатів дослідження показала значний комерційний потенціал розробленого програмного забезпечення, включаючи прогнозування витрат на виконання наукової роботи та впровадження її результатів, а також комерційних ефектів від реалізації.

Таким чином, дана робота зробила вагомий внесок у підвищення рівня захищеності IoT-систем, запропонувавши впровадження моделі нульової довіри та алгоритму динамічного встановлення довіри. Отримані результати мають значну теоретичну та практичну цінність, можуть бути використані для покращення безпеки існуючих IoT-систем та розробки нових захищених рішень, а також стануть корисними для подальших наукових досліджень у цій галузі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. AVSystem - Shaping the world of connected devices. AVSystem – Shaping The World of Connected Devices. URL: https://www.avsystem.com/coiote-iot-device-management-platform/iot-device-management/.

2. BasuMallick C. What Is IoT Device Management? Definition, Key Features, and Software - Spiceworks. Spiceworks Inc. URL: https://www.spiceworks.com/tech/iot/articles/what-is-iot-device-management/.

3. Gillis A. S. What is IoT (Internet of Things) and How Does it Work? | Definition from TechTarget. IoT Agenda. URL: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT.

4. IoT Device Management_IoT_IoTDM-HUAWEI CLOUD. Huawei Cloud. URL: https://www.huaweicloud.com/intl/en-us/product/iot.html.

5. Revolutionary IoT Device Management | Shaping the future IoT. Novatec Consulting GmbH. URL: https://www.novatec-gmbh.de/en/insights/blog/iot-device-management/.

6. thingsboard. ⊳ IoT Device Management Platform. ThingsBoard. URL: https://thingsboard.io/device-management/.

7. What is IoT Device Management? | Definition & Benefits | Software AG. Software AG. URL: https://www.softwareag.com/en_corporate/resources/iot/article/iot-device-management.html.

8. What is IoT Device Management? | NinjaOne. NinjaOne. URL: https://www.ninjaone.com/blog/what-is-iot-device-management/.

9. What is IoT? - Internet of Things Explained - Amazon AWS. Amazon Web Services, Inc. URL: https://aws.amazon.com/ru/what-is/iot/#:~:text=The%20term%20IoT,%20or%20Internet,as%20between%20the%20devices%20themselves.

10. What is IoT? The internet of things explained. Network World. URL: https://www.networkworld.com/article/963923/what-is-iot-the-internet-of-things-explained.html.

11. What is the Internet of Things (IoT)? | IBM. IBM - United States. URL: https://www.ibm.com/topics/internet-of-things.

12. What is the Internet of Things (IoT)?. Oracle | Cloud Applications and Cloud Platform. URL: https://www.oracle.com/jo/internet-of-things/what-is-iot/.

13. What Is the Internet of Things (IoT)? With Examples. Coursera. URL: https://www.coursera.org/articles/internet-of-things.

14. What is the IoT? Everything you need to know about the Internet of Things right now. ZDNET. URL: https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/.

15. Cloudflare Application Services | Security and Performance. Cloudflare. URL: https://www.cloudflare.com/application-services/?utm_source=google&utm_medium=cpc&utm_campaign=DG_EMEA_ENG_G_Search_Generic_Beta_Applications-Security&utm_content=Beta_Generic_Applications-Security_Core&utm_term=cyber+security&campaignid=71700000112716322&adgroupid=58700008486001012&creativeid=662071359069&gclid=Cj0KCQiA67CrBhC1ARIsACKAa8Q17aOUR4pjkbG6p-6JfA6-aLhFR_-ig7BqZ7VtZrN6wUzk1Nhp6nkaAgwEEALw_wcB&gclsrc=aw.ds).

16. Кібербезпека. URL: https://www.span.eu/ua/рішення-та-послуги/сервіси-з-безпеки/кібербезпека/?gad_source=1&gclid=Cj0KCQiA67CrBhC1ARIsACKAa8REXQS8JOs2ZaaOW6g5OyaeVWM8ksz3M6viwjX_pclBefnGsQgibfMaAgYHEALw_wcB.

17. Contributors to Wikimedia projects. Information security - Wikipedia. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/Information_security

18. Imperva. URL: https://www.imperva.com/learn/data-security/information-security-infosec/.

19. What is information security (infosec)?. Cisco. URL: https://www.cisco.com/c/en/us/products/security/what-is-information-security-infosec.html.

20. What is information security (infosec)? Goals, types and applications. Exabeam. URL: https://www.exabeam.com/explainers/information-security/information-security-goals-types-and-applications/.

21. Yasar K., Wright G., Teravainen T. What is information security (infosec)? – techtarget definition. Security. URL: https://www.techtarget.com/searchsecurity/definition/information-security-infosec.

22. What is information security? | IBM. IBM in Deutschland, Österreich und der Schweiz | IBM. URL: https://www.ibm.com/topics/information-security.

23. What is information security? - geeksforgeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/what-is-information-security/.

24. INFOSEC - Glossary | CSRC. NIST Computer Security Resource Center | CSRC. URL: https://csrc.nist.gov/glossary/term/INFOSEC.

25. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.

26. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця : ВНТУ, 2016. 113 с.

27. What is authorization? - examples and definition - auth0. *Auth0*. URL: https://auth0.com/intro-to-iam/what-is-authorization .

28. Academy B. Seed Phrase | Binance Academy. Binance Academy. URL: https://academy.binance.com/en/glossary/seed-phrase.

29. What is authorization? - examples and definition - auth0. *Auth0*. URL: https://auth0.com/intro-to-iam/what-is-authorization .

30. What is javascript? A basic introduction to JS for beginners. Hostinger Tutorials. URL: https://www.hostinger.com/tutorials/what-is-javascript).

31. Visual studio code. Visual Studio Code. URL: https://code.visualstudio.com/).

32. Редактор коду visual studio code. Habr. URL: https://habr.com/ua/articles/490754

33. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript.

34. Electron. Electron. URL: https://www.electronjs.org/ .

35. Node.js. Node.js. URL: https://nodejs.org/uk .

36. The Modern JavaScript. URL: https://javascript.info/ .

37. Sufiyan T. What is node.js: a comprehensive guide. *Simplilearn.com*. URL: https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs).

38. Megida D. What is javascript? A definition of the JS programming language. freeCodeCamp.org. URL: https://www.freecodecamp.org/news/what-is-javascript-definition-of-js/

39. What is javascript? A basic introduction to JS for beginners. Hostinger Tutorials. URL: https://www.hostinger.com/tutorials/what-is-javascript).

40. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця : ВНТУ, 2021. 42 с.

41. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця : ВНТУ, 2016. 113 с

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції "Управління інформаційною безпекою" кафедри МБІС
д.т.н., професор
_____ Юрій ЯРЕМЧУК
"___" _____ 2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:
Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри
08-72.МКР.000.00.000.ТЗ

Керівник магістерської кваліфікаційної роботи
к.т.н., доцент каф. МБІС
Грицак А.В

Вінниця – 2024 р.

1. Найменування та область застосування

Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри **2.Підстава для розробки**

Розробка виконується на основі наказу ректора ВНТУ № 81 від 11. 03. 2024 р. **3. Мета та призначення розробки**

3.1 Мета розробки: підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри

3.2 Призначення: підвищення захищеності системи моніторингу та управління IoT-пристроями

4. Джерела розробки

4.1. Ахрамович В. М. Ідентифікація й аутентифікація, керування доступом // Сучасний захист інформації. – 2016. №4.– С. 47-51.

4.2. Бурячок В.Л. Політика інформаційної безпеки: підручник. / В.Л.Бурячок, Р.В.Грищук, В.О.Хорошко / За заг. ред. докт. техн. наук, проф. В.О. Хорошка. – К.: ПВП «Задруга», 2014. – 222 с.

4.3. Єсін В.І. Безпека інформаційних систем і технологій / В.І.Єсін, О.О. Кузнецов, Л.С. Сорока. – Харків: ХНУ імені В.Н. Каразіна, 2013. – 632 с.

4.4. ZakariaOmar, ZangooeiToomaj, MohdAfiziMohdShukran. Enhancing Mixing Recognition-Based and Recall-Based Approach in Graphical Password Scheme. IJACT, Vol. 4, No. 15, pp. 189-197, 2012.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний засіб повинен виконувати процес автентифікації користувачів у системі.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Pentium 1500 МГц і подібні до них;

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6.1 Обов'язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.4

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1.	Визначення напрямку магістерської роботи, формулювання теми	1.03.2024	15.03.2024	
2.	Аналіз предметної області обраної теми	15.03.2024	1.04.2024	
3.	Розробка роботи	1.04.2024	10.04.2024	
4.	Написання магістерської роботи на основі розробленої теми	10.04.2024	20.04.2024	
5.	Передзахист магістерської кваліфікаційної роботи	20.04.2024	10.05.2024	
6.	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	10.05.2024	4.06.2024	
7.	Захист магістерської кваліфікаційної роботи	10.06.2024	10.06.2024	

10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;
– програмний додаток;
– презентація;
– відзив керівника роботи;
– відзив опонента

Технічне завдання до виконання прийняв _____ Ратушняк С.С

Додаток Б. Лістинг програми

```
import machine
import time
import urequests
# Налаштування сенсора температури (наприклад, DHT11)
sensor = machine.Pin(2, machine.Pin.IN)
dht_sensor = dht.DHT11(sensor)
# URL сервера для передачі даних
url = "http://your_server_address/api/sensor_data/"
while True:
    try:
        dht_sensor.measure()
        temperature = dht_sensor.temperature()
        humidity = dht_sensor.humidity()

        # Формування даних для передачі
        data = {
            "temperature": temperature,
            "humidity": humidity
        }

        # Відправка даних на сервер
        response = urequests.post(url, json=data)
        print(response.text)
        # Затримка перед наступним вимірюванням
        time.sleep(60)
    except Exception as e:
        print("Error:", e)
        time.sleep(60)
from django.db import models

class SensorData(models.Model):
    timestamp = models.DateTimeField(auto_now_add=True)
    temperature = models.FloatField()
    humidity = models.FloatField()

    def __str__(self):
        return f"Temp: {self.temperature}, Humidity: {self.humidity} at {self.timestamp}"
from rest_framework import serializers
from .models import SensorData

class SensorDataSerializer(serializers.ModelSerializer):
    class Meta:
        model = SensorData
        fields = ['timestamp', 'temperature', 'humidity']
from rest_framework import viewsets
from .models import SensorData
from .serializers import SensorDataSerializer

class SensorDataViewSet(viewsets.ModelViewSet):
    queryset = SensorData.objects.all()
from django.urls import path, include
from rest_framework.routers import DefaultRouter
from .views import SensorDataViewSet

router = DefaultRouter()
```

```
router.register(r'sensor_data', SensorDataViewSet)

urlpatterns = [
   path('api/', include(router.urls)),
```

settings.py:

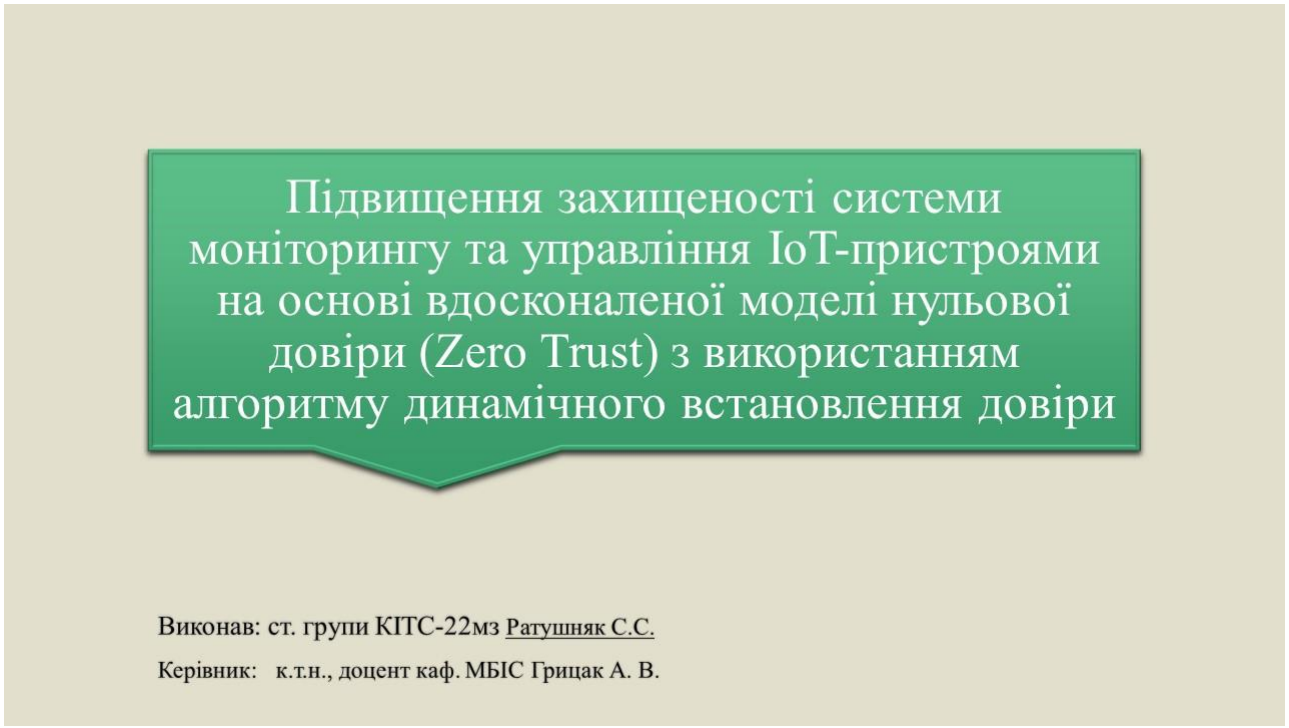
```
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.postgresql',
      'NAME': 'your_database_name',
      'USER': 'your_database_user',
      'PASSWORD': 'your_database_password',
      'HOST': 'localhost',
      'PORT': '5432',
   }
import React, { useEffect, useState } from 'react';
import axios from 'axios';
const SensorData = () => {
   const [data, setData] = useState([]);
   useEffect(() => {
      axios.get('/api/sensor_data/')
         .then(response => {
            setData(response.data);
         })
         .catch(error => {
            console.error("There was an error fetching the data!", error);
         });
   }, []);
   return (
      <div>
         <h1>Sensor Data</h1>
         <table>
            <thead>
               <tr>
                  <th>Timestamp</th>
                  <th>Temperature</th>
                  <th>Humidity</th>
               </tr>
            </thead>
            <tbody>
               {data.map((item) => (
                  <tr key={item.timestamp}>
                     <td>{item.timestamp}</td>
                     <td>{item.temperature}</td>
                     <td>{item.humidity}</td>
                  </tr>
               ))}
            </tbody>
         </table>
      </div>
   );
};
export default SensorData;
access_policies = {
   'admin': ['read', 'write', 'delete'],
   'user': ['read', 'write'],
   'guest': ['read']
}
```

```
        # Функція перевірки доступу
        def check_access(role, action):
            if action in access_policies.get(role, []):
                return True
            return False
        import jwt
        import datetime
        SECRET_KEY = 'your_secret_key'
        # Функція створення JWT-токена
        def create_token(user_id, role):
            payload = {
                'user_id': user_id,
                'role': role,
                'exp': datetime.datetime.utcnow() + datetime.timedelta(hours=1)
            }
            token = jwt.encode(payload, SECRET_KEY, algorithm='HS256')
            return token

        # Функція перевірки JWT-токена
        def verify_token(token):
            try:
                payload = jwt.decode(token, SECRET_KEY, algorithms=['HS256'])
                return payload
            except jwt.ExpiredSignatureError:
                return None
        def evaluate_trust(user_id, device_id, action):
            trust_score = 100  # Початковий рівень довіри
            if action == 'suspicious_activity':
                trust_score -= 50

            # Оцінка довіри на основі попередньої історії
            history = get_user_device_history(user_id, device_id)
            for record in history:
                if record['action'] == 'failed_authentication':
                    trust_score -= 10
            return trust_score
def get_user_device_history(user_id, device_id):
            # Отримання даних з бази даних або логів
            history = [
                {'timestamp': '2024-01-01 12:00:00', 'action': 'successful_authentication'},
                {'timestamp': '2024-01-01 12:30:00', 'action': 'failed_authentication'}
            ]
            return history
        mport logging
        logging.basicConfig(filename='audit.log', level=logging.INFO)
        def log_activity(user_id, device_id, action, result):
            log_message = f"User: {user_id}, Device: {device_id}, Action: {action}, Result: {result}"
            logging.info(log_message)
        log_activity('user1', 'device1', 'login', 'success')
```

Додаток В. Ілюстративний матеріал



Актуальність та мета

Актуальність

- Зі збільшенням кількості IoT-пристроїв та їх ролі в критичних інфраструктурах зростають ризики кіберзагроз і вразливостей. Традиційні методи захисту не здатні ефективно справлятися з динамічними і розподіленими природою IoT-мереж. Вдосконалена модель нульової довіри (Zero Trust) з алгоритмом динамічного встановлення довіри пропонує більш надійний підхід, забезпечуючи постійний контроль і адаптивну перевірку кожного запиту на доступ, що значно підвищує безпеку систем моніторингу та управління IoT.

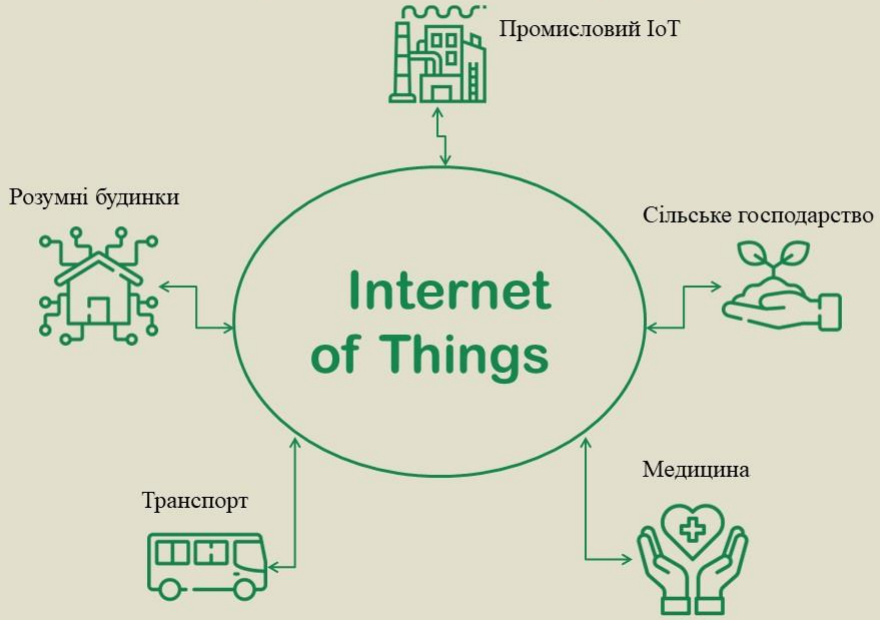
Мета

- Метою є розробка і впровадження вдосконаленої моделі нульової довіри для систем моніторингу та управління IoT-пристроями, яка використовує алгоритм динамічного встановлення довіри для підвищення захищеності. Цей підхід забезпечить постійний контроль доступу, адаптивність до змін у середовищі та ефективний захист від сучасних кіберзагроз, що дозволить зберегти цілісність і безпеку IoT-інфраструктури.

IoT-пристрої

IoT-пристрої (Internet of Things) – це фізичні об'єкти, обладнані сенсорами, програмним забезпеченням, та іншими технологіями, які підключаються до Інтернету або інших мереж для збору та обміну даними. Вони можуть включати побутові прилади, автомобілі, медичне обладнання, промислові машини та багато інших пристроїв, що забезпечують автоматизацію, контроль та оптимізацію різних процесів і функцій в реальному часі.

Використання IoT пристроїв



Zero Trust

Zero Trust (модель нульової довіри) – це підхід до безпеки інформаційних систем, який передбачає, що жодному користувачу, пристрою або системі не можна довіряти за замовчуванням, незалежно від їхнього місцезнаходження в мережі. Кожен запит на доступ перевіряється і авторизується в реальному часі, а доступ надається лише на мінімально необхідному рівні, що знижує ризики витоків і кіберзагроз. Модель базується на принципах: "не довіряй нікому" і "перевіряй все".

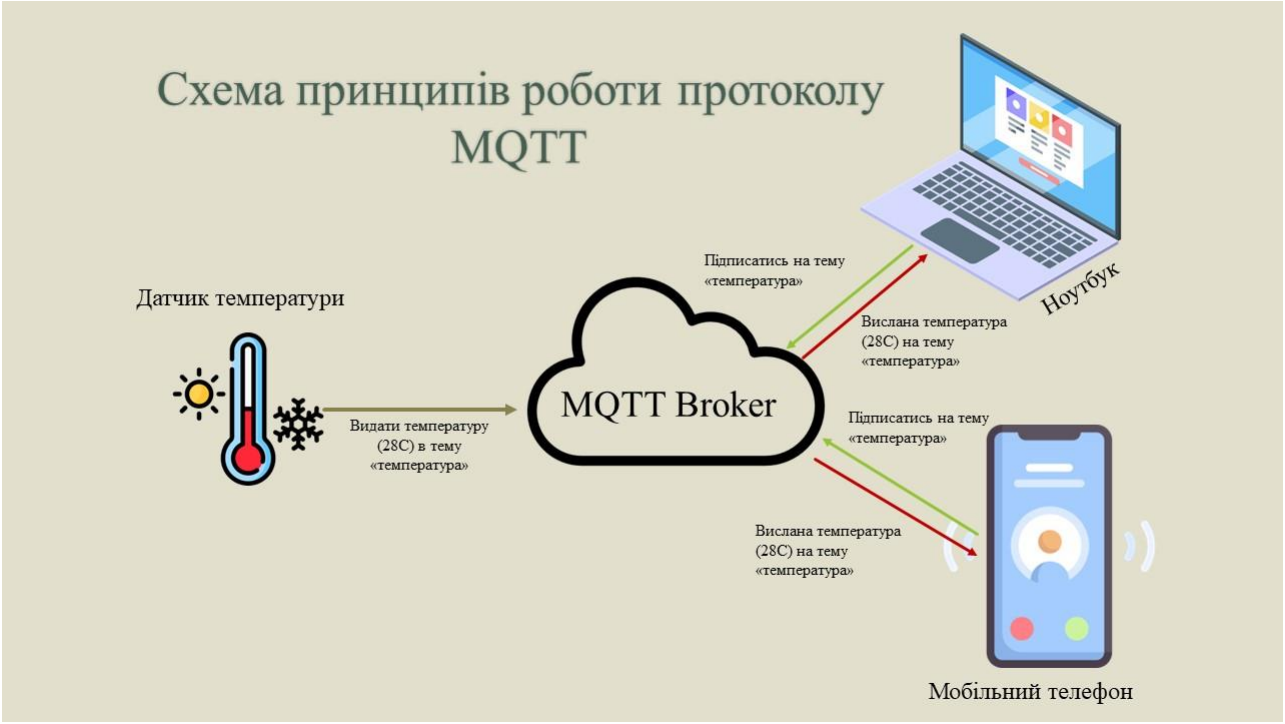
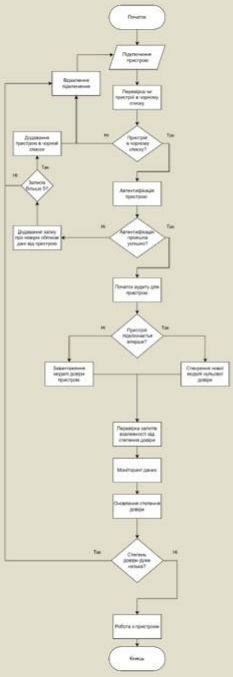


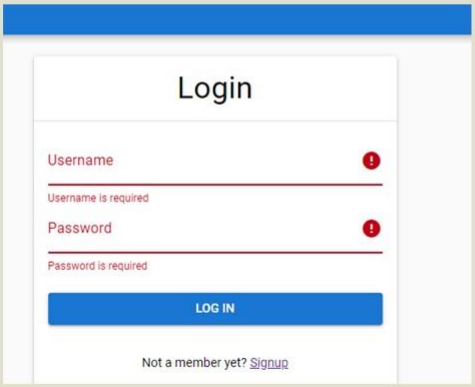
Схема принципів роботи протоколу MQTT
Датчик температури
Видати температуру (28C) в тему «температура»
MQTT Broker
Підписатись на тему «температура»
Вислана температура (28C) на тему «температура»
Ноутбук
Підписатись на тему «температура»
Вислана температура (28C) на тему «температура»
Мобільний телефон

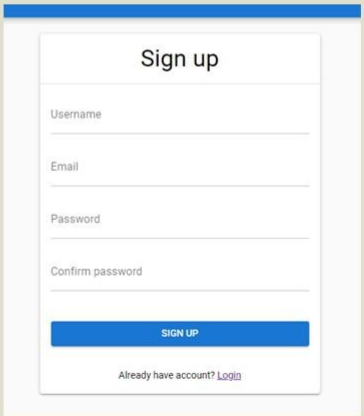

Схема принципів роботи протоколу CoAP
Мережа Інтернет
Мережа пристроїв
HTTP
HTTP
Сервер
CoAP
CoAO – пристрій
CoAO – проксі
CoAP
CoAO – пристрій
Сервер



Алгоритм підвищення захищеності системи моніторингу та управління IoT-пристроями

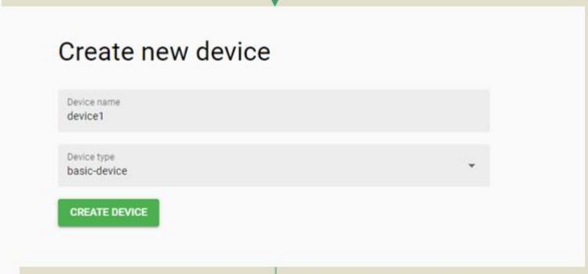
Інтерфейс додатку





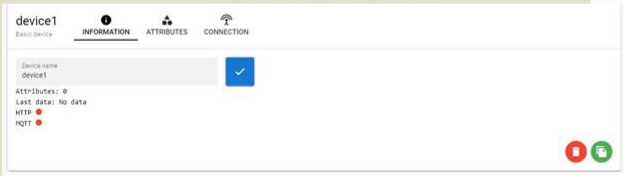
Інтерфейс додатку

Список IoT пристроїв



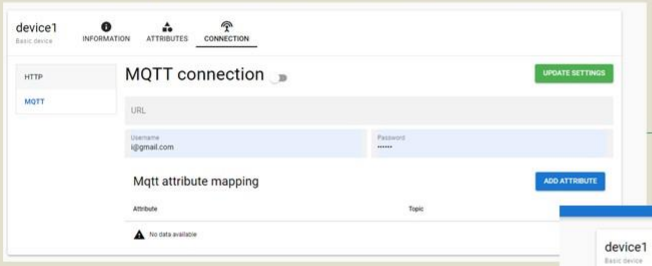
Стоірнка додавання IoT пристроя

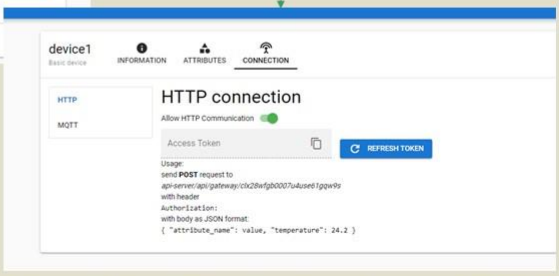
Сторінка меню пристрою



Інтерфейс додатку

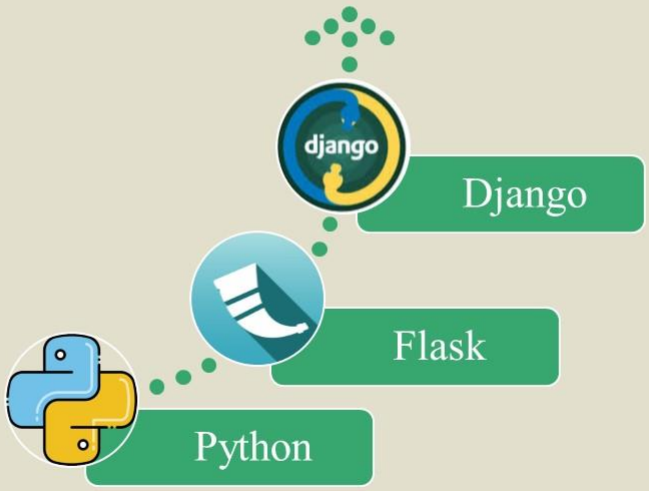
Стоірнка надання дозволу на підключення





Надання доступу пристрою через протокол HTTP

ВИКОРИСТАНІ ТЕХНОЛОГІЇ



ВИСНОВОК

Вдосконалення захищеності систем моніторингу та управління IoT-пристроями через використання моделі нульової довіри (Zero Trust) з алгоритмом динамічного встановлення довіри забезпечує суттєве підвищення рівня безпеки. Цей підхід дозволяє ефективно виявляти та запобігати несанкціонованим доступам і кіберзагрозам, забезпечуючи адаптивний контроль доступу на основі постійної перевірки довіреності пристроїв та користувачів. Така модель відмовляється від традиційних концепцій периметрової безпеки, спрямовуючи увагу на захист кожного елементу мережі в реальному часі. Це значно підвищує стійкість IoT-інфраструктури, робить її гнучкою до змін у середовищі загроз і готовою до викликів, які постають перед сучасними інформаційними системами.

ДЯКУЮ ЗА УВАГУ

Додаток Г. Протокол перевірки на антиплагіат

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ
ЗАПОЗИЧЕНЬ

Назва роботи: Підвищення захищеності системи моніторингу та управління IoT-пристроями на основі вдосконаленої моделі нульової довіри (Zero Trust) з використанням алгоритму динамічного встановлення довіри

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 98 % Схожість 2 %

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ (підпис) Коваль Н.П. (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ (підпис) Ратушняк С.С. (прізвище, ініціали)

Керівник роботи _____ (підпис) Грицак А.В. (прізвище, ініціали)