Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2025 р.

# ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
**за другим (магістерським) рівнем вищої освіти**

на тему

**“*Дослідження та реалізація апаратно-програмних комплексів для виявлення атак на рівні IoT-пристроїв з використанням квантованих нейромереж.*”**



Виконав здобувач вищої освіти
ІІ курсу, групи КІ-24М
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Прокопено Є.С.
«____» _____ 2025 р.

Керівник проекту
кандидат технічних наук, доцент
_____ Дрєєв О.М.
«____» _____ 2025 р.
Рецензент _____
_____

м. Кропивницький

# АНОТАЦІЯ

**Прокопенко Є.С. Дослідження та реалізація апаратно-програмних комплексів для виявлення атак на рівні IoT-пристроїв з використанням квантованих нейромереж. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2025.**

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для автономного виявлення та протидії кібератакам у мережах Інтернету речей (IoT) за допомогою технологій Edge AI.

Метою розробки є підвищення рівня захищеності мереж IoT шляхом розробки та реалізації апаратно-програмного комплексу виявлення вторгнень на основі адаптованих легковажих нейромереж.

Об'єктом дослідження є процеси функціонування та обміну даними в мережах інтернету речей в умовах кібернетичного впливу та ресурсних обмежень апаратної платформи.

Предметом дослідження є методи та апаратно-програмні засоби нейромережевого виявлення аномалій, оптимізовані для виконання на мікроконтролерах.

Методи дослідження базуються на методах системного аналізу для визначення вимог до системи, теорії штучних нейронних мереж для побудови моделі класифікації трафіку, математичній статистиці та методах розробки вбудованого програмного забезпечення.

Результат роботи – апаратно-програмна реалізація системи виявлення атак на базі мікроконтролера ESP32-S3 з використанням квантованих нейронних мереж.

В процесі роботи над програмною моделлю виконано аналіз існуючих систем захисту IoT-інфраструктури, обґрунтовано вибір апаратної платформи з підтримкою векторних інструкцій та методів компресії моделей TinyML. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача у вигляді адміністративної панелі на мові Python для візуалізації статистики та керування вузлами. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на мікроконтролерах серії ESP32-S3 під управлінням ОС FreeRTOS та на ПК з ОС Windows 10/11 для моніторингу мережі.

Програма розроблена в середовищі Visual Studio Code з використанням фреймворку ESP-IDF та бібліотеки TensorFlow Lite for Microcontrollers.

**Ключові слова:** інтернет речей, кібербезпека, нейронні мережі, ESP32-S3, TinyML, виявлення вторгнень, Edge AI.

# ABSTRACT

**Prokopenko Y.S. Research and implementation of hardware and software complexes for detecting attacks at the IoT device level using quantized neural networks. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2025.**

In this final qualification work for the second (master's) level of higher education, software has been developed, which is intended for the autonomous detection and counteraction of cyberattacks in Internet of Things (IoT) networks using Edge AI technologies.

The purpose of the development is to increase the security level of IoT networks by developing and implementing a hardware and software intrusion detection system based on adapted lightweight neural networks.

The object of the research is the processes of functioning and data exchange in Internet of Things networks under cyber influence and resource constraints of the hardware platform.

The subject of the research is methods and hardware-software tools for neural network anomaly detection optimized for execution on microcontrollers.

The research methods are based on systems analysis methods to define system requirements, the theory of artificial neural networks for traffic classification modeling, mathematical statistics, and embedded software development methods.

The result of the work is a hardware and software implementation of an attack detection system based on the ESP32-S3 microcontroller using quantized neural networks.

In the process of working on the software model, an analysis of existing IoT infrastructure protection systems was performed, and the choice of a hardware platform with vector instruction support and TinyML model compression methods was justified. All components of the developed software are fully described.

A user-friendly user interface has been developed in the form of an administrative panel in Python for visualizing statistics and managing nodes. Instructions for working with the software tools are provided.

The program can be used on ESP32-S3 series microcontrollers running FreeRTOS and on PCs with Windows 10/11 for network monitoring.

The program was developed in the Visual Studio Code environment using the ESP-IDF framework and the TensorFlow Lite for Microcontrollers library.

**Keywords:** internet of things, cybersecurity, neural networks, ESP32-S3, TinyML, intrusion detection, Edge AI.

ЗМІСТ

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

| | | |
|---|---|---|
| ОС | – | Операційна система |
| ПЗ | – | Програмне забезпечення |
| AI | – | Basic Input-Output System |
| MQTT | – | Message Queuing Telemetry Transport |
| PSRAM | – | Pseudo Static Random Access Memory |
| SIMD | – | Single Instruction, Multiple Data |

# ВСТУП

**Актуальність теми.** Швидке зростання кількості підключених пристроїв у глобальній мережі Інтернет речей (IoT) призводить до стрімкого збільшення векторів кібератак. Різноманітність пристроїв та використання спрощених протоколів обміну даними роблять цей сегмент вразливим до несанкціонованого втручання, що підтверджується зростанням кількості інцидентів, пов'язаних із ботнетами та DDoS-атаками [1, 2].

Для запобігання таких інцидентів необхідно захищати кінцеві вузли, які функціонують в умовах жорстких апаратних обмежень (енергоспоживання, обчислювальна потужність, обсяг пам'яті). Традиційні засоби захисту, такі як класичні системи виявлення вторгнень (IDS) або антивірусне програмне забезпечення, не можуть бути ефективно розгорнуті на мікроконтролерах через високі вимоги до ресурсів системи.

Враховуючи що перенесення функцій аналізу безпеки у хмарне середовище створює деяккі затримки в реакції на інциденти та залежність від стабільності каналу зв'язку. Перспективним напрямом вирішення цієї проблеми є концепція Edge AI (граничний штучний інтелект), що передбачає виконання алгоритмів машинного навчання безпосередньо на кінцевому пристрої. Застосування оптимізованих легковагих нейронних мереж дозволяє забезпечити автономне виявлення аномалій у мережевому трафіку в реальному часі.

Таким чином, дослідження та реалізація апаратно-програмного комплексу для виявлення атак на рівні IoT-пристроїв з використанням легковагих нейромереж є актуальною науково-прикладною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі.

**Мета й завдання дослідження.** Метою роботи є підвищення рівня захищеності мереж IoT шляхом розробки та реалізації апаратно-програмного комплексу виявлення вторгнень на основі адаптованих ейзеносних нейромереж.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

1. Провести огляд існуючих систем виявлення вторгнень в IoT та методів розгортання нейромереж на мікроконтролерах.

2. Обґрунтувати вибір апаратної платформи та програмних засобів для реалізації граничних обчислень.

3. Розробити структурну та функціональну схеми комплексу виявлення атак на базі вибраного мікроконтролера.

4. Здійснити програмну реалізацію модуля інференсу нейронної мережі з використанням векторних інструкцій для аналізу трафіку в реальному часі.

5. Провести експериментальне дослідження ефективності розробленого комплексу при виявленні типових мережевих атак.

**Об'єктом дослідження** є процеси функціонування та обміну даними в мережах інтернету речей в умовах кібернетичного впливу та ресурсних обмежень апаратної платформи.

**Предметом дослідження** є методи та апаратно-програмні засоби нейромережевого виявлення аномалій, оптимізовані для виконання на мікроконтролерах.

**Методи дослідження.** У данній науковій роботі використані такі методи аналізу, а саме методи системного аналізу для визначення вимог до системи захисту, теорія штучних нейронних мереж для побудови моделі класифікації трафіку, математична статистика для оцінки точності виявлення атак, експериментальне дослідження для перевірки часових характеристик та ресурсоємності розробленого ПЗ.

**Наукова новизна отриманих результатів.** У процесі розв'язання завдань отримано наступні результати:

1) Удосконалено метод виявлення мережевих атак на рівні кінцевих пристроїв шляхом адаптації нейронної мережі до специфіки векторних інструкцій

мікроконтролера ESP32-S3, що дозволило зменшити час обробки при збереженні точності класифікації.

2) Набув подальшого розвитку підхід до побудови автономних шлюзів безпеки, який, на відміну від існуючих хмарних рішень, забезпечує локальну ізоляцію скомпрометованих вузлів без необхідності звернення до центрального сервера.

**Практична цінність отриманих результатів** полягає у створенні діючого макету апаратно-програмного комплексу, здатного працювати в мережах які складаються з абсолютно різних пристроїв та контролерів. Розроблені алгоритми та програмне забезпечення дозволяють інтегрувати модуль захисту в існуючі системи «розумного будинку» та промислового інтернету речей з мінімальними витратами на апаратну частину. Достовірність результатів підтверджена даними експериментального тестування на реальному обладнанні.

# 1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

## 1.1 Призначення системи

Розроблена система призначена для забезпечення захисту периферійних вузлів мереж інтернету речей шляхом виявлення та блокування мережевих атак у режимі реального часу. Вона функціонує як повністю автономний шлюз безпеки, що аналізує весь інтернет трафік безпосередньо на межі мережі (від кінцеаого вузла з датчиком до керуючого всією системою контролера), усуваючи необхідність передачі чутливих даних до хмарних серверів [3].

Основою функціонування комплексу є технологія TinyML (Tiny Machine Learning), яка дозволяє розгортати моделі машинного навчання на мікроконтролерах з обмеженими обчислювальними ресурсами. На відміну від традиційних сигнатурних систем виявлення вторгнень (IDS), що потребують регулярного оновлення баз даних відомих загроз, запропоноване рішення використовує нейромережевий підхід для виявлення аномалій, що дозволяє ідентифікувати нові та модифіковані типи атак.

Ключовим елементом системи є нейронна мережа на базі архітектури CNN (Convolutional Neural Network), оптимізована за допомогою квантування для адекватної роботи з мікроконтролером ESP32-S3. Використання даного мікроконтролера обумовлено наявністю набору векторних інструкцій, що прискорюють обчислення тензорів, необхідних для інференсу нейромережі [4].

Функціональне призначення комплексу включає:

— моніторинг бездротового ефіру (Wi-Fi) в режимі Promiscuous Mode для перехоплення пакетів даних;

— попередню обробку вхідного трафіку та формування векторів ознак;

— класифікацію мережевої активності на «нормальну» та «аномальну»;

— активну протидію атакам шляхом відправки деавторизаційних фреймів або блокування трафіку на рівні шлюзу;

— генерацію сповіщень тривоги через захищений канал зв’язку.

### 1.2 Область застосування

Областю застосування розробленого комплексу є сегменти мереж Інтернету речей, що характеризуються високими вимогами до приватності даних, низькою затримкою реакції та обмеженою пропускною здатністю каналів зв’язку. Система орієнтована на використання в умовах гетерогенного середовища, де одночасно функціонують пристрої різних виробників із різними протоколами обміну даними (MQTT, CoAP, HTTP).

Впровадження комплексу доцільне у наступних сферах:

1) Розумний будинок: захист побутових IoT-пристроїв (камери, термостати, замки), які часто не мають вбудованих механізмів захисту та рідко отримують оновлення прошивки від виробника. Комплекс виступає як проміжний вузол, що фільтрує трафік перед його потраплянням у локальну мережу.

2) Промисловий Інтернет речей (IIoT): моніторинг роботи датчиків та механізмів у виробничих зонах, де відсутнє стабільне підключення до Інтернету. Локальна обробка даних дозволяє уникнути затримок, критичних для технологічних процесів.

3) Критична інфраструктура: забезпечення безпеки ізольованих частин мережі, де використання інших IDS заборонено політиками безпеки.

Для розгортання системи необхідна наявність бездротової мережі стандарту IEEE 802.11 b/g/n. Апаратна реалізація базується на мікроконтролері ESP32-S3, що дозволяє інтегрувати модуль захисту безпосередньо в існуюче мережеве обладнання або використовувати його як окремий пристрій.

Програмна складова комплексу розроблена з використанням фреймворку ESP-IDF (Espressif IoT Development Framework) та операційної системи FreeRTOS, що забезпечує цілеспрямоване та передбачувальне виконання завдань збору та аналізу даних. Адміністративний інтерфейс реалізовано як веб-додаток, доступний через

локальну мережу, що дозволяє налаштовувати параметри чутливості нейромережі та переглядати журнали подій.

# 2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

## 2.1 Огляд існуючих систем, технологій, архітектур та програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Сучасний розвиток концепцій периферійних обчислень призвів до появи класу пристроїв, здатних виконувати алгоритми штучного інтелекту безпосередньо на «межі» мережі. Для реалізації системи виявлення вторгнень на рівні IoT-пристроїв важливим є вибір оптимальної архітектури пристрою, яка буде забезпечувати баланс між обчислювальною потужністю, енергоефективністю та вартістю.

**Аналіз апаратних платформ для реалізації Edge AI**

Вибір апаратної платформи для реалізації концепції граничних обчислень обумовлений необхідністю досягнення балансу між обчислювальною потужністю, енергоефективністю та вартістю кінцевого виробу. Специфіка задачі це виявлення аномалій у мережевому трафіку в реальному часі. Така задача висуває жорсткі вимоги до архітектури мікроконтролера, зокрема до наявності апаратної підтримки векторних операцій та пропускної здатності підсистеми пам’яті.

**Архітектурні особливості ESP32-S3**

В основу розроблюваного комплексу покладено ESP32-S3, побудований на базі двох 32-бітних ядер Xtensa LX7 із тактовою частотою до 240 МГц. Архітектура LX7 належить до класу RISC (Reduced Instruction Set Computer) з 5-7 ступеневим конвеєром, що дозволяє досягти високої щільності коду та показника продуктивності на рівні 600 DMIPS[5].

Ключовою перевагою даної архітектури в контексті задач машинного навчання є розширений набір інструкцій, що включає підтримку векторних операцій (SIMD – Single Instruction, Multiple Data). На відміну від скалярних обчислень, де одна інструкція обробляє одну пару операндів, SIMD-інструкції

дозволяють виконувати операції над 128-бітними векторами даних за один тактовий цикл.

Це важливо для операцій матричного множення та згортки, які складають до 90% обчислювального навантаження при роботі з нейромережами. Використання спеціалізованих інструкцій (наприклад, для прискорення операцій *Multiply-Accumulate* – MAC) дозволяє значно знизити показник CPI (Clock cycles per instruction) при обробці тензорів ваг нейромережі. Для моделей, квантованих до форматів int8 або int16, приріст продуктивності порівняно зі стандартними ALU (Arithmetic Logic Unit) може досягати 3-5 разів залежно від типу шару нейромережі.

Крім того, ядра LX7 мають конфігуровані локальні пам'яті інструкцій та даних, що дозволяє розміщувати необхідні ділянки коду (наприклад, обробник переривань мережевого інтерфейсу) та буфери даних безпосередньо в низьколатентній області, мінімізуючи затримки вибірки.

**Порівняльний аналіз альтернативних платформ**

Для обґрунтування вибору ESP32-S3 проведено порівняльний аналіз із конкурентними рішеннями в класі вбудованих систем.

**Kendryte K210 (Архітектура RISC-V)**

Цей чіп позиціонується як спеціалізований AI-прискорювач. Він містить два 64-бітних ядра RISC-V та апаратний блок KPU (Knowledge Processing Unit), призначений для прискорення операцій згортки (Conv2D), активації та пулінгу. KPU дозволяє виконувати інференс CNN (наприклад, YOLO) з високою енергоефективністю.

Щодо недоліків то K210 сильно оптимізований під задачі комп'ютерного зору і має обмежену гнучкість для обробки одновимірних часових рядів, характерних для аналізу мережевого трафіку. Найбільшим обмеженням є складність інструментарію розробки та фрагментованість документації, що підвищує ризики інтеграції з мережевими стеками реального часу.

**Raspberry Pi Pico (RP2040, Архітектура ARM Cortex-M0+)**

Двоядерний процесор із частотою 133 МГц є привабливим через низьку вартість та доступність. Однак архітектура Cortex-M0+ є найбільш простою в лінійці ARM, має конвеєр з 2 стадій та не містить апаратного блоку FPU або DSP-інструкцій.

З недоліків можна віднести відсутність SIMD-інструкцій, яка призводить до необхідності програмної емуляції операцій над векторами, що дуже збільшує час обробки. Обсяг оперативної пам'яті (264 КБ) є недостатнім для одночасного зберігання буферів мережевих пакетів, стеку TCP/IP та тензорів нейромережі. В умовах реального аналізу трафіку RP2040 не здатний забезпечити потрібний для данних задач час обробки та реакції, що призводить до втрати пакетів [7].

**Вимоги до підсистеми пам'яті та проблема розміщення моделі**

Ефективність Edge AI системи прямо залежить від пропускної здатності пам'яті, оскільки операції з нейромережою вимагають постійного завантаження ваг моделі з енергонезалежної пам'яті в оперативну.

Основною проблемою при розгортанні нейромереж на мікроконтролерах є обмежений обсяг внутрішньої оперативної пам'яті. Для ESP32-S3 цей обсяг становить 512 КБ, з яких частина (приблизно 150-200 КБ) резервується під потреби Bluetooth та Wi-Fi стеків та операційної системи FreeRTOS. Корисного простору, що залишається для тензорів та проміжних обчислень, часто недостатньо для моделей складніших за простий перцептрон.

Для вирішення цієї проблеми використовується зовнішня оперативна пам'ять. У конкурентних рішеннях (наприклад, STM32H7) часто застосовується зовнішня SDRAM із паралельним інтерфейсом, що забезпечує високу швидкість, але потребує значної кількості ліній GPIO (до 30-40).

ESP32-S3 використовує альтернативний підхід – підхід з псевдо оперативною пам'ятю (PSRAM) яка підключена через послідовний інтерфейс. У порівнянні з попередніми поколіннями, що використовували чотири лінії для швидкої передачі данних, обраний модуль ESP32-S3-WROOM-1 підтримує інтерфейс який

використовує вісім ліній для передачі та обробки даних. Це дозволяє передавати 8 біт даних за одиницю часу, що подвоює теоретичну пропускну здатність порівняно з минулими поколіннями. Хоча латентність доступу (це затримка між моментом, коли система отримує запит на дані чи команду, і моментом, коли ці дані стають доступними або операція завершується) до PSRAM вища, ніж до внутрішньої SRAM (через накладні витрати протоколу SPI та кешування), механізм блоку керування пам'ятю дозволяє прозоро відображати до 32 МБ зовнішньої пам'яті в адресний простір процесора, що знімає жорсткі обмеження на розмір моделі нейромережі.

Таким чином, архітектура ESP32-S3 з підтримкою векторних інструкцій та PSRAM є оптимальним компромісом, що забезпечує достатню продуктивність для задач класифікації трафіку за умови застосування методів оптимізації моделі.

**Методи оптимізації нейромереж для вбудованих систем**

Розгортання глибоких нейронних мереж (DNN) на мікроконтролерах ESP32-S3 пов'язане з подоланням обмежень щодо обсягу енергонезалежної пам'яті для зберігання ваг та оперативної пам'яті для зберігання проміжних тензорів активації. Для адаптації моделей використовується комплекс методів компресії, серед яких ключовими є квантування та прунінг.

**Квантування**

Квантування є найбільш ефективним методом зменшення розмірності моделі та прискорення обчислень на апаратних платформах, що підтримують SIMD-інструкції. Суть методу полягає у відображенні неперервного діапазону значень ваг та активацій, представлених у форматі з плаваючою комою (FP32, 32-біт), у дискретний набір значень фіксованої розрядності, як правило, 8-бітних цілих чисел (INT8).

Перехід до цілочисельної арифметики дозволяє зменшити обсяг пам'яті, необхідної для зберігання моделі, у 4 рази, а також значно знизити енергоспоживання процесора, оскільки операції над цілими числами енергетично менш витратні, ніж операції з плаваючою комою [12].

Математично процес рівномірного афінного квантування описується формулою відновлення (de-quantization):

$$r = S(q - Z) \tag{2.1}$$

де:

— r — апроксимоване значення дійсного числа (Real value);

— S (Scale) — масштабний коефіцієнт (додатне число формату FP32), що визначає крок дискретизації;

— q — квантоване значення, q [q_min, q_max] (для INT8 діапазон від -128 до 127);

— Z (Zero-point) — ціле число, що відповідає дійсному значенню 0. Цей параметр є важливим для коректного виконання операцій доповнення нулями (zero-padding) у згорткових шарах та коректної роботи функцій активації типу ReLU.

В контексті розробки системи на базі ESP32-S3 буде доцільним розглянути розглядаються два підходи до квантування:

1) Пост-тренувальне квантування. Процедура виконується після завершення навчання моделі. Ваги округлюються до найближчих цілих значень, а параметри S та Z калібруються на репрезентативній вибірці даних. Перевагою методу є швидкість імплементації, проте можлива деградація точності детектування атак, особливо при значному динамічному діапазоні ваг.

2) Навчання з урахуванням квантування**.** Під час навчання до обчислювального графу додаються вузли, що емулюють помилку квантування. Це дозволяє нейромережі адаптувати ваги таким чином, щоб мінімізувати втрату точності при подальшому переході до INT8. Для задач, де критичною є точність та повнота, застосування QAT є більш доцільним [13].

Варто зазначити що вибір формату INT8 для ESP32-S3 обумовлений архітектурою ядра Xtensa LX7, яка містить спеціалізовані інструкції для прискорення 8-бітних векторних операцій, що забезпечує значний приріст продуктивності порівняно з програмною емуляцією FP32.

**Прунінг**

Прунінг, або «проріджування» нейромережі, базується на гіпотезі, що значна частина параметрів глибоких мереж є надлишковою. Процес полягає у виявленні та видаленні синаптичних зв'язків (ваг), абсолютне значення яких наближається до нуля, оскільки їх вплив на вихідний сигнал нейрона є нехтуваним.

Розрізняють два види прунінгу:

— Неструктурований: Обнулення окремих ваг у довільному порядку. Це призводить до утворення розріджених матриць (sparse matrices). Хоча це дозволяє досягти високого ступеня стиснення, ефективна реалізація розріджених обчислень на мікроконтролерах загального призначення є складною через нерегулярний доступ до пам'яті та відсутність апаратної підтримки розрідженої алгебри.

— Структурований: Видалення цілих структурних елементів мережі — каналів згортки, фільтрів або нейронів у повнозв'язних шарах. Такий підхід зберігає щільну структуру матриць (dense matrices), що дозволяє використовувати стандартні оптимізовані бібліотеки лінійної алгебри та векторизацію на ESP32-S3 без додаткових накладних витрат [14].

**Особливості програмної реалізації у TensorFlow Lite for Microcontrollers**

Для роботи з нейромережами на мікроконтролерах використовують спеціалізований фреймворк TensorFlow Lite for Microcontrollers (TFLM). Його ключовою відмінністю від інших рішень є специфічна модель роботи з оперативною пам'яттю, спрямована на забезпечення стабільності та детермінованості системи.

У традиційних операційних системах пам'ять для об'єктів виділяється динамічно (функції malloc/new), що в умовах тривалої роботи пристрою призводить до фрагментації. Для мікроконтролерів, які функціонують у безперервно без перезавантаження, це створює ризик непередбачуваних збоїв.

TFLM вирішує цю проблему за допомогою концепції Tensor Arena. Це заздалегідь виділений неперервний блок пам'яті, розмір якого визначається на етапі компіляції або ініціалізації. Всі вхідні, вихідні та проміжні тензори моделі

розміщуються всередині цієї «арени». Планувальник пам’яті фреймворку автоматично оптимізує розміщення тензорів, повторно використовуючи ділянки пам’яті для тензорів, час життя яких не перетинається [15]. Такий підхід гарантує відсутність фрагментації та дозволяє точно розрахувати необхідний обсяг RAM ще до етапу прошивки пристрою.

**Висновки**

Завдяки проведеному аналізу існуючих систем, апаратних платформ та методів реалізації штучного інтелекту мікроконтролерах дозволяє сформулювати наступні висновки:

1) Існуючі системи виявлення вторгнень хмарного базування не задовольняють вимогам мереж IoT щодо латентності та автономності, а класичні сигнатурні методи є неефективними на мікроконтролерах через обмежену кількість пам’яті.

2) Оптимальною платформою для реалізації автономного вузла захисту є ESP32-S3, архітектура якого поєднує енергоефективність з підтримкою векторних інструкцій для прискорення нейромережевих обчислень та наявністю інтерфейсу PSRAM для роботи з об’ємними даними.

3) Для виявлення аномалій у мережевому трафіку в умовах невизначеності профілю атак найбільш перспективною є архітектура автоенкодера або одновимірної згорткової мережі (1D-CNN), що навчається в режимі «без учителя».

4) Необхідною умовою реалізації системи на мікроконтролері є застосування методів компресії моделі нейромережі, зокрема квантування до формату INT8, що дозволяє задіяти специфічні апаратні можливості (наприклад DSP-інструкції) та забезпечити роботу моделі в межах виділеної Tensor Arena без використання динамічної пам’яті.

Таким чином, для подальшої розробки обрано гібридний підхід, що передбачає побудову основого вузла на базі ESP32-S3 з використанням квантованої нейромережі для аналізу трафіку в реальному часі.

## 2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

**Вибір мов програмування**

Проектування та реалізація поставленого в роботі завдання вимагає застосування гібридного підходу до розробки програмного забезпечення. Даний підхід базується на чіткому розділенні життєвого циклу нейромережевої моделі на два етапи, що мають принципово різні вимоги до обчислювальних середовищ: етап навчання та етап виконання.

Етап навчання передбачає роботою з великими масивами даних, необхідністю швидкого прототипування архітектури мережі та використанням потужних графічних прискорювачів для навчання моделі. Натомість, етап виконання відбувається безпосередньо на вибраному мікроконтролері ESP32-S3 в умовах жорстких обмежень щодо пам’яті, енергоспоживання та вимог до часу реакції. Враховуючи цю різноманітність, для реалізації проєкту було обрано пару мов програмування, а саме Python для етапу з початковим навчанням моделі та C/C++ для етапу розгортання нейромережі на мікроконтролері.

**Обґрунтування використання Python для розробки моделі**

Мова програмування Python, починаючи з версії 3.8+, обрана як основний інструмент для підготовки даних, конструювання архітектури нейронної мережі та її навчання. Вибір обумовлений розповсюдженням Python у сфері машинного навчання та наявністю розвиненої екосистеми бібліотек, що забезпечують ефективну роботу з будь якими обчисленнями які можуть знадобитися в роботі.

Ключові компоненти задіяні в роботі:

1) NumPy: Ця бібліотека є фундаментом для наукових обчислень у Python. Вона надає підтримку багатовимірних масивів та матриць, а також велику колекцію високорівневих математичних функцій. Конче важливим є те, що обчислювальне ядро NumPy реалізовано на мовах C та Fortran, що дозволяє виконувати векторні та

матричні операції (основу лінійної алгебри нейромереж) з продуктивністю, близькою до нативного коду [16].

2) Pandas: Якщо проанаізувати IoT-трафік, вхідні дані (журнали пакетів, часові мітки, розміри корисного навантаження) представляють собою часові ряди. Бібліотека Pandas забезпечує ефективні механізми для індексації, зрізу, агрегації та ресемплінгу часових рядів, що необхідно для формування вхідних векторів ознак для нейромережі.

3) Keras/TensorFlow API: Використання API Keras дозволяє абстрагуватися від низькорівневої реалізації графу обчислень та зосередитися на побудові моделі нейромережі (кількість шарів, типи нейронів, функції активації). Це значно прискорює процес пошуку оптимальної топології та гіперпараметрів навчання.

Важливо зазначити, що Python у даному проєкті використовується виключно як інструментальний засіб. Отримані результати (ваги моделі та структура графу) конвертуються у формат, незалежний від мови програмування Python, для подальшого використання.

**Обґрунтування використання C/C++ для реалізації прошивки**

Для програмної реалізації вузла аналізу (Secure Gateway) на базі мікроконтролера ESP32-S3 обрано мову **C++** ,а саме стандарт ISO/IEC 14882:2017, відомий як C++17, з використанням елементів мови C для взаємодії з апаратними абстракціями.

Залучення інтерпретованих мов (наприклад, MicroPython або JavaScript) для задач виявлення вторгнень у реальному часі є недоцільним через низку технічних обмежень, які вирішуються при переході на компільовані мови C/C++:

1) Детермінованість та керування пам’яттю. Найбільший керованих мов ( у контексті данної роботи) є наявність автоматичного збирача сміття. Процес звільнення пам’яті є недетермінованим і може викликати непередбачувані затримки у виконанні основного коду. Для системи детекції вторгнень, яка повинна обробляти вхідний потік мережевих пакетів близької до швидкості каналу зв’язку, будь-яка затримка призведе до переповнення буферів прийому та втрати пакетів,

що зробить систему «сліпою» до атак у цей момент часу. C++ дозволяє повністю контролювати життєвий цикл об'єктів. У розробці використовується парадигма RAII (Resource Acquisition Is Initialization) та статичне виділення пам'яті для критичних буферів (зокрема, Tensor Arena для нейромережі), що гарантує відсутність фрагментації купи та стабільний час виконання циклу обробки [17].

2) Низькорівневий доступ до апаратних ресурсів. Ідея данної роботи передбачає інтенсивну взаємодію з периферією мікроконтролера, зокрема з контролером Wi-Fi та таймерами. Мова C++ забезпечує можливість прямого доступу до фізичних адрес регістрів керування ними через вказівники, що необхідно для налаштування режимів роботи радіомодуля (наприклад, переведення у Promiscuous Mode). Крім того, ефективна обробка потоку даних вимагає використання механізму прямого доступу до пам'яті (DMA – Direct Memory Access). Налаштування дескрипторів DMA та робота з кільцевими буферами в оперативній пам'яті можлива лише засобами системних мов програмування, що забезпечують роботу з сирими байтами та адресною арифметикою без накладних витрат віртуальних машин.

3) Продуктивність та оптимізація. Компілятор xtensa-esp32-elf-g++ (частина ESP-IDF Toolchain) здійснює трансляцію коду C++ у машинний код оптимізований для архітектури Xtensa, виконуючи глибоку оптимізацію на етапі компіляції (рівень -O2 або -O3). Така оптимізація включає інлайнінг функцій, розгортання циклів та, що найважливіше, використання спеціалізованих DSP-інструкцій процесора ESP32-S3. Бібліотека TensorFlow Lite for Microcontrollers, написана на C++11 і використовує шаблони для генерації оптимізованих ядер операцій нейромережі, що дозволяє уникнути накладних витрат на динамічне розподілення викликів під час роботи нейромережі.

4) Сумісність з іншими пристроями. Основний фреймворк розробки для чіпів Espressif (ESP-IDF) – базується на мові C. Використання C++ дозволяє непомітно та без зайвих витрат інтегрувати системні виклики операційної системи FreeRTOS

та мережевого стека LwIP, забезпечуючи при цьому вищий рівень абстракції та модульності коду завдяки об’єктно-орієнтованому підходу [18].

Таким чином, комбінація Python для етапу навчання та C++17 для етапу виконання є оптимальним технологічним рішенням, що дозволяє поєднати гнучкість наявного інструментарію з високою продуктивністю та надійністю вбудованого програмного забезпечення.

**Середовище розробки та інструментарій**

Враховуючи різноманітну природу проектованої системи, що поєднує компоненти низькорівневого програмування мікроконтролерів та високорівневого моделювання нейронних мереж, було сформовано спеціалізований набір інструментальних засобів.

**Інтегроване середовище Visual Studio Code з розширенням PlatformIO**

Для розробки вбудованого вузла аналізу на базі ESP32-S3 було вибрано Visual Studio Code з обов’язковим використанням плагіну PlatformIO.

Незважаючи на свою простоту та відносно низький поріг входження Arduino IDE, має низку критичних обмежень для розробки данного проєкту інженерного рівня: відсутність повноцінних механізмів налагодження, примітивний редактор коду без можливостей рефакторингу та глобальна система керування бібліотеками, що унеможливлює гарантування відтворюваності збірки на інших робочих станціях.

Натомість, вибір PlatformIO як ключового плагіну обумовлений наступними технічними перевагами:

1. Декларативне керування залежностями та конфігурацією**.** Ключовим елементом проекту в екосистемі PlatformIO є конфігураційний файл platformio.ini. Цей файл дозволяє зафіксувати параметри збірки, що є необхідними для забезпечення стабільної роботи нейромережевих алгоритмів. Зокрема, механізм керування бібліотеками дозволяє вказати не лише назву необхідного компонента, але і його точну версію або хеш комміту в репозиторії Git. Наприклад, директива lib_deps = TensorFlowLite_ESP32 @ 1.0.0 гарантує, що компілятор використає саме

потирібну для нас версію бібліотеки, на якій можливо проводилося тестування системи. Це вирішує проблему коли оновлення сторонньої бібліотеки призводить до непрацездатності проекту, наприклад через зміни в API [22]. Крім того, файл platformio.ini дозволяє визначати параметри тактування процесора (board_build.f_cpu = 240000000L), розмір виділеної пам'яті під стек та купу, а також розділи таблиці розділів Flash-пам'яті, що необхідно для розміщення великих вагових коефіцієнтів нейромережі.

2. Інтелектуальне редагування коду. Ще одною позитивною рисою VS code є Language Server Protocol. Він забезпечує статичний аналіз коду C/C++ у реальному часі. Такий функціонал надає можливості контекстного автодоповнення, швидкої навігації до визначень функцій та класів, а також перегляду документації до API безпосередньо в редакторі. Для проекту, що використовує складні структури даних, бібліотеки TensorFlow Lite та низькорівневі виклики FreeRTOS, це дозволяє значно знизити кількість синтаксичних та логічних помилок ще до етапу компіляції. Інтеграція з автоматичними аналізаторами коду (наприклад, Clang-Tidy) забезпечує дотримання стандартів кодування та виявлення деяких потенційних проблем з безпекою пам'яті [23].

3. Апаратне налагодження. Одним із вирішальних факторів вибору PlatformIO є вбудована підтримка уніфікованого інтерфейсу налагодження. На відміну від методу «printf-debugging» (виведення логів у послідовний порт), який вносить часові затримки та змінює поведінку системи реального часу, використання апаратного налагоджувача через вбудований USB-JTAG у чіпі ESP32-S3 дозволяє:

— Виконувати покрокове виконання коду для аналізу логіки прийняття рішень нейромережею.

— Встановлювати точки зупинки (Breakpoints), в тому числі умовні та по даним (Watchpoints).

— Переглядати вміст регістрів процесора Xtensa LX7, стеку викликів (Call Stack) та довільних ділянок оперативної пам'яті без зупинки периферійних

модулів. Це дозволяє аналізувати стан буферів мережевих пакетів безпосередньо в момент виявлення атаки.

Важливо підкреслити, що PlatformIO виступає в ролі системи автоматизації збірки та керування проектом, тоді як «фундаментом» є Espressif IoT Development Framework. Вибір нативної версії ESP-IDF (на відміну від шару абстракції Arduino Core) дозволяє отримати повний контроль над апаратними ресурсами мікроконтролера.

Використання ESP-IDF надає доступ до:

— Операційної системи реального часу FreeRTOS: ЇЇ архітектура базується на основі незалежних задач з різними пріоритетами (наприклад, задача захоплення трафіку має вищий пріоритет за задачу відправки телеметрії) та використання примітивів синхронізації (семафори, черги, м'ютекси) забезпечує безпечний обмін даними між ядрами процесора.

— Драйверів периферії: Цей фреймворк надає гнучкий низькорівневий доступ до Wi-Fi стека в режимі Promiscuous Mode, керування апаратними таймерами та сторожовим таймером (Watchdog Timer) для забезпечення відмовостійкості системи.

— Оптимізованих бібліотек: Можливо використати бібліотеки esp-dsp для виконання цифрової обробки сигналів та матричних операцій з використанням SIMD-інструкцій процесора [24].

**Середовище для навчання моделі**

Етап розробки, навчання та валідації нейромережевої моделі вимагає значних обчислювальних потужностей, які часто перевищують можливості персональних комп'ютерів. Для вирішення цієї задачі використано хмарне середовище Google Colaboratory.

Це середовище, побудоване на базі Jupyter Notebooks, дозволяє виконувати код мовою Python у хмарі з доступом до різних апаратних прискорювачів. Вибір Google Colab обґрунтований наступними факторами:

— Доступ до GPU**:** данний вибір дозволяє використання графічних процесорів класу NVIDIA Tesla T4 або A100 для розпаралелення операцій множення матриць, що є основою алгоритму зворотного поширення помилки. Це скорочує час навчання моделі з декількох годин до десятків хвилин, що дозволяє швидко перевіряти різні гіпотези та оптимізувати деякі параметри мережі.

— Попередньо налаштоване оточення: середовище містить встановлені бібліотеки TensorFlow, Keras, NumPy, Pandas та Scikit-learn, що мінімізує час на налаштування робочого оточення.

— Візуалізація даних: інтерактивні можливості Jupyter Notebook дозволяють будувати графіки процесу навчання, матриці плутанини та візуалізувати розподіл ознак мережевого трафіку, що необхідно для аналізу якості датасету [25].

Після завершення навчання в Google Colab виконується конвертація моделі у формат TensorFlow Lite (.tflite) та її трансляція у масив байтів C++ (C-byte array) для інтеграції у прошивку мікроконтролера.

**Протоколи обміну даними в розподіленій IoT-системі**

Специфіка виявлення вторгнень у реальному часі формує набір критичних вимог до транспортного рівня: мінімізація затримки доставки повідомлень, зниження енергоспоживання радіомодулів та забезпечення цілісності даних за нявності перешкод у бездротовому каналі зв’язку.

**Аналіз застосовності протоколу HTTP**

Архітектура REST (тип веб-застосунку) через HTTP-зв’язок є непрактичною для цієї системи, оскільки вона має кілька технічних обмежень. Модель Request-Response вимагає від клієнта встановлення зв’язку з кожною транзакцією, що робить асинхронне надсилання керуючих команд сервером до пристрою неможливим без використання ресурсомістких методів, таких як Long Polling та WebSockets. [26].

Формат заголовків HTTP створює надлишкове навантаження на канал зв’язку, тому що передача метаданих ( наприклад, User-Agent, Content-Type, Authentication) збільшує розмір пакету на сотні байт, нівелюючи ефективність передачі корисного

навантаження обсягом у декілька байт (код події, статус сенсора). Для пристроїв з батарейним живленням така надлишковість призводить до прискореного розряду акумулятора через збільшений час активності трансивера.

**Обґрунтування вибору протоколу MQTT**

В якості стандарту обміну даними обрано протокол MQTT (Message Queuing Telemetry Transport) версій 3.1.1/5.0, що базується на асинхронній моделі «Публікація/Підписка». Центральний елемент архітектури — це брокер повідомлень (MQTT Broker) який виконує функцію маршрутизатора, забезпечуючи повне розділення відправника та отримувача, що дозволяє їм обмінюватися повідомленями без прямого знання про один одного. Вузол аналізу публікує події безпеки у відповідні топіки, не володіючи інформацією про стан та кількість підписників , тобто панелей моніторингу.

Технічні характеристики та переваги реалізації:

1) Мінімізація трафіку. Бінарна структура протоколу забезпечує фіксований заголовок розміром 2 байти. Зниження накладних витрат дозволяє ефективно використовувати смугу пропускання зашумлених каналів Wi-Fi (2.4 ГГц) та зменшує вірогідність колізій пакетів у мережі [27].

2) Рівні якості обслуговування (QoS – Quality of Service). Специфікація MQTT регламентує три рівні гарантії доставки, вибір яких залежить від критичності даних:

— QoS 0 (At most once): передача без підтвердження. Застосовується для потокової телеметрії (завантаження CPU, температура чіпа), втрата окремих пакетів якої не впливає на загальну картину.

— QoS 1 (At least once): гарантована доставка з підтвердженням (PUBACK). Відправник зберігає повідомлення у локальному буфері до отримання квитанції від брокера. У разі таймауту виконується повторна відправка з прапором DUP. Цей рівень імплементовано для передачі алертів про атаки, забезпечуючи надійність сповіщення навіть при короткочасних розривах з’єднання.

— QoS 2 (Exactly once): доставка точно один раз через чотирьохетапне рукостискання. Використання обмежено критичними керуючими командами через високі затримки [28].

3) Моніторинг доступності вузлів (LWT). Механізм «Last Will and Testament» дозволяє автоматизувати виявлення аварійного відключення вузла. При встановленні з'єднання клієнт реєструє на брокері повідомлення LWT. У випадку некоректного розриву TCP-сесії (втрата живлення, апаратний збій, глушіння сигналу атакуючим) брокер автоматично публікує це повідомлення. Система моніторингу інтерпретує подію як інцидент доступності.

**Захист каналу передачі даних**

Передача даних у відкритому вигляді (clear text) створює вектори атак типу Man-in-the-Middle (MitM) та пасивного перехоплення (Sniffing). Забезпечення конфіденційності реалізовано шляхом інкапсуляції MQTT у захищений тунель TLS/SSL (MQTTS) на порту 8883.

На стороні мікроконтролера ESP32-S3 виконується верифікація сертифіката сервера (CA Certificate) для запобігання підміні брокера. Криптографічні операції (шифрування AES-128/256, хешування SHA-256) прискорюються апаратним криптомодулем SoC, що мінімізує вплив шифрування на загальну продуктивність системи.

**Висновки**

Обґрунтування вибору засобів розробки базується на вимогах до продуктивності, енергоефективності та надійності проектованої системи захисту IoT. нами було прийнято рішеня щодо використання мови Python та бібліотек (NumPy, TensorFlow) які забезпечуть ефективне моделювання та навчання нейромережі на етапі підготовки.

Щодо реалізації прошивки, то мова C++ (ISO/IEC 14882:2017) надає необхідний рівень контролю над апаратними ресурсами ESP32-S3, забезпечуючи детермінованість часових характеристик та роботу з пам'яттю без фрагментації.

Поєднання середовища Visual Studio Code та плагіну PlatformIO гарантує нам відтворюваність збірки та надає професійні засоби налагодження, перевершуючи можливості Arduino IDE. Базовий фреймворк ESP-IDF забезпечує доступ до функцій операційної системи реального часу FreeRTOS.

Комунікація буде відбуватися за допомогою протоколу MQTT, а саме QoS 1 у поєднанні з шифруванням TLS. Це оптимальним рішенням для передачі критичних сповіщень у системі, забезпечуючи баланс між надійністю доставки та використанням ресурсів каналу зв'язку.

Обрана конфігурація засобів дозволяє реалізувати автономний шлюз безпеки, здатний функціонувати в режимі реального часу з мінімальними затримками щодо реакції на інциденти.

### 2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає апаратно-програмний комплекс, який призначено для виявлення та протидії атакам на рівні IoT-пристроїв з використанням легковагих нейромереж.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів захисту IoT-інфраструктури для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи виявлення аномальної поведінки трафіку та нейтралізації загроз в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи (вбудоване ПЗ мікроконтролера та серверну частину), що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про виявлені кіберінциденти, аномалії мережевої активності та статус ізоляції скомпрометованих вузлів;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

# 3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

## 3.1 Опис функціонування системи

### Ініціалізація комплексу та конфігурація пам’яті

При своєму запуску система ініціює виконання завантажувача другого рівня. Для цього важливим етапом ініціалізації є розгортання карти пам’яті відповідно до попередньо визначеної таблиці розділів. Це потрібно задля стабільності роботи нейромережі та можливості оновлення прошивки «по повітрю».Нами була використана схема розмітки  Flash-пам’яті на 4 мегабайти яка наведена в таблиці 3.1.

Таблиця 3.1 – Розподіл простору Flash-пам’яті

| Тип | Підтип | Зсув | Розмір | Призначення |
|---|---|---|---|---|
| data | nvs | 0x9000 | 20 КБ | Енергонезалежне сховище налаштувань |
| data | phy_init | 0xe000 | 4 КБ | Калібрувальні дані радіомодуля |
| app | factory | 0x10000 | 1.5 МБ | Основний образ прошивки |
| app | ota_0 | 0x180000 | 1.5 МБ | Слот для оновлення прошивки |
| data | coredump | 0x300000 | 64 КБ | Збереження дампу пам’яті при критичних збоях |

Після монтування файлової системи виконується налаштування таймерів для забезпечення відмовостійкості. Враховуючи високе навантаження на процесор під час роботи нейромережі нами було запроваджено таку систему контролю:

1. Interrupt Watchdog – це апаратний таймер, що гарантує перемикання контексту переривань. Тайм-аут встановлено на рівні 300 мс для запобігання блокуванню системи драйвером Wi-Fi.

2. Task Watchdog – являє собою програмний механізм FreeRTOS, що відстежує стан задач. Задача нейромережевого аналізу повинна періодично скидати таймер, підтверджуючи нормальне функціонування алгоритму [29].

Далі відбувається переміщення статичного буфера Tensor Arena у зовнішній модуль PSRAM. Таким чином використання зовнішньої пам'яті через високошвидкісний інтерфейс Octal SPI дозволяє виділити неперервний блок розміром до 2 МБ для зберігання вхідних, вихідних та проміжних тензорів, що унеможливлює помилки викликані розподіломресурсів під час роботи системи.

Після ініціалізації стека TCP/IP та встановлення захищеного з'єднання з MQTT-брокером, радіомодуль переводиться у режим прослуховування.

**Цикл збору даних та структура кадрів**

Механізм щодо бору даних реалізований нами на механіки зворотних викликів, що спрацьовують при виявлені підходящого фрейму стандарту IEEE 802.11. Функція обробки promiscuous_rx_cb отримує доступ до сирого пакету, структура якого відповідає стандарту 802.11 MAC Header загальною довжиною 24 байти (без врахування поля QoS Control).

Ключові поля заголовка, що підлягають аналізу:

— Frame Control (2 байти): містить ідентифікатори версії протоколу, типу (Management, Control, Data) та підтипу (Beacon, Probe Request, Deauthentication) кадру. Також включає прапори ToDS та FromDS, що визначають напрямок руху пакету відносно точки доступу.

— Duration/ID (2 байти): визначає час, на який середовище передачі буде зайнято. Аномально великі значення в цьому полі можуть свідчити про спробу атаки «Virtual Jamming» (Nav Attack).

— Address Fields (6 байт x 3): MAC-адреси приймача (DA), передавача (SA) та точки доступу (BSSID).

Для буферизації потоку даних було створено структуру кільцевого буфера фіксованої ємності N. Керування доступом здійснюється за допомогою двох індексів з яких Head (голова) для запису нових даних та Tail (хвіст) для читання

нейромережевим модулем. Розрахунок індексу для запису наступного елемента розраховується за формулою:

$$Index_{next} = (Index_{current} + 1) \, mod \, N \quad (3.1)$$

Для запобігання пошкодженню даних, що знаходяться в процесі аналізу, застосовано стратегію «Drop on Full». У випадку, коли буфер заповнений (умова (Head + 1) mod N == Tail), нові пакети ігноруються до моменту звільнення місця задачею аналізу. Це гарантує цілісність часового вікна, яке вже сформовано та передано на вхід нейромережі.

**Нейромережевий аналіз**

Основною логікою виявлення аномалій являється обробка вектора ознак, сформованих з сирих даних буфера. Цей процес називається розрахунком інженерії ознак яка передбачає перетворення статистичних характеристик трафіку в числовий вектор V.

Для детектування складних атак обчислюються наступні метрики: ентропія Шеннона, середньоквадратичне відхилення інтервалів прибуття, співвідношення керуючих кадрів ($R_{mgmt}$).

Використання ентропії Шеннона щодо розмірів пакетів потрібна для виявлення тунелювання даних або аномального розподілу навантаження.

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_2 P(x_i) \, , \quad (3.2)$$

де $P(x_i)$ - ймовірність появи пакету розміром $x_i$ у поточному вікні. Низька ентропія характерна для DDoS-атак, висока - для нормального трафіку.

Середньоквадратичне відхилення інтервалів прибуття дозволяє виявити автоматизовані інструменти сканування, які генерують пакети з фіксованою періодичністю, на відміну від стохастичної природи людської активності.

Співвідношення керуючих кадрів це частка кадрів управління до загальної їх кількості. Різке зростання $R_{mgmt}$ свідчить про атаки типу Deauthentication Flood.

Перед подачею на вхідний шар нейромережі вектор ознак підлягає квантуванню. Оскільки модель TensorFlow Lite оптимізована для цілочисельних

обчислень (int8), дійсні значення ознак $x_{float}$ (формат float32) перетворюються у цілі числа $x_{quant}$ за формулою афінного квантування:

$$x_{quant} = clamp\left(\left\lceil\frac{x_{float}}{S}\right\rceil + Z, -128{,}127\right), \quad (3.3)$$

де S (Scale) - масштабний коефіцієнт, Z (Zero-point) - зміщення нуля, визначені на етапі навчання моделі. Операція clamp обмежує значення діапазоном 8-бітного знакового цілого. Такий підхід зменшує вимоги до пам'яті для зберігання ваг моделі з 4 байт до 1 байта на параметр, що є необхідним для обмеженої кількості ресурсів до будь якого мікроконтролера [30].

**Механізм реагування на інциденти**

Логіка прийняття рішень та керування режимами роботи системи реалізована у вигляді скінченного автомата. Визначено наступні стани системи:

— STATE_IDLE: початкова ініціалізація, очікування підключення до Wi-Fi/MQTT.

— STATE_MONITOR: пасивне накопичення даних у кільцевий буфер.

— STATE_ANALYSIS: активна фаза інференсу нейромережі.

— STATE_ALERT: формування та відправка сповіщення про загрозу.

— STATE_ISOLATION: виконання активних контрзаходів (блокування).

Перехід у стан STATE_ALERT відбувається за умови, що ймовірність атаки P(Attack), розрахована нейромережею, перевищує порогове значення Threshold. Для запобігання перевантаженню каналу зв'язку та перенасичення одноманітними повідомленнями впроваджено алгоритм.

Алгоритм обмежує частоту відправки повідомлень про один і той самий тип атаки. Нове повідомлення генерується лише за умови виконання нерівності:

де $\Delta T_{cooldown}$— період «охолодження» (наприклад, 5000 мс). Якщо атака триває, система лише інкрементує лічильник інцидентів у локальному лозі, але не ініціює нову транзакцію MQTT, доки не спливе час затримки.

У стані STATE_ISOLATION система може відправляти спеціально сформовані пакети (наприклад, Channel Switch Announcement) для міграції пристроїв на інший частотний канал, ізолюючи їх від джерела перешкод.

### 3.2 Розробка структурної схеми комплексу

Архітектура системи декомпозована на функціональні рівні, що забезпечують збір, первинну обробку, аналіз та передачу інформації.

**Обґрунтування топології мережі**

Мережева архітектура реалізована за топологією типу «Зірка». Роль центрального концентратора та арбітра безпеки виконує інтелектуальний шлюз. Зовнішні вузли (сенсори, актуатори) функціонують як клієнти, трафік яких підлягає моніторингу та аналізу.

Застосування топології згаданої вище обумовлено необхідністю мінімізації часових затримок при передачі тривожних сповіщень. На відміну від децентралізованих мереж, де маршрутизація пакетів здійснюється через ланцюжок проміжних вузлів, централізована схема виключає затримки при ретрансляції. Такий підхід є необхідним для системи виявлення вторгнень, де час реакції на атаку має складати мілісекунди.

Фізичним середовищем для передачі даних є радіоканал стандарту IEEE 802.11n (Wi-Fi) у діапазоні 2.4 ГГц.

**Апаратна реалізація центрального вузла**

Основою обчислювального ядра є високоефективна система ESP32-S3-WROOM-1-N16R8. Архітектура мікроконтролера включає два 32-розрядних процесори Xtensa LX7, що працюють на тактовій частоті 240 МГц. Вибір даної платформи обумовлений наявністю розширеного набору векторних інструкцій, що забезпечують апаратне прискорення операцій матричного множення та згортки, які складають основу обчислювального навантаження згорткових нейронних мереж.

**Підсистема пам’яті та інтерфейс Octal SPI**

Критичною особливістю обраної модифікації модуля (N16R8) є розширена підсистема пам’яті, що включає:

1. Flash-пам’ять складає 16 МБ і підключена через інтерфейс Quad SPI. Вона використовується для зберігання прошивки, файлової системи LittleFS та статичних вагових коефіцієнтів нейромережі.

2. Оперативна пам’ять (зокрема PSRAM) складає 8 МБ і підключена через високошвидкісний інтерфейс Octal SPI ( надалі OPI).

Використання OPI PSRAM є визначальним проектним рішенням. На відміну від стандартної внутрішньої SRAM (512 КБ), якої недостатньо для розміщення складних моделей глибокого навчання, зовнішня пам’ять дозволяє виділити значний обсяг адресного простору під Tensor Arena - безперервну ділянку пам’яті для зберігання вхідних/вихідних тензорів та проміжних результатів активації нейронів. Інтерфейс OPI забезпечує передачу даних по 8 лініях за один такт, що гарантує високу пропускну здатність шини, необхідну для динамічного завантаження шарів моделі без суттєвих затримок інференсу.

Додатково, наявність 8 МБ оперативної пам’яті дозволяє реалізувати глибокий кільцевий буфер (Ring Buffer) для захоплення мережевих пакетів. Це забезпечує режим моніторингу Zero-drop sniffing, нівелюючи ризик втрати пакетів під час пікових навантажень на центральний процесор при обробці результатів нейромережі.

**Зовнішні інтерфейси та підсистема реєстрації**

Для забезпечення автономності функціонування та реалізації режиму «Чорної скриньки», структурна схема передбачає підключення зовнішнього накопичувача — карти пам’яті MicroSD. Обмін даними реалізовано через апаратний інтерфейс SPI (режим VSPI). Накопичувач використовується для ведення журналів подій безпеки та збереження дампів аномального трафіку для подальшого криміналістичного аналізу.

Комунікація із зовнішнім світом здійснюється через вбудований радіомодуль Wi-Fi, який підтримує роботу в режимі прослуховування. Цей режим потрібен для захоплення кадрів канального рівня без підключення до конкретної точки доступу. Передача телеметрії на сервер здійснюється через захищений канал MQTTS.

**Структурна схема комплексу**

Графічне представлення архітектури системи, що ілюструє логічні зв'язки та високошвидкісні інтерфейси обміну даними між основними функціональними блоками, наведено на рисунку 3.1.

Запропонована схема, завдяки використанню розширеної пам'яті OPI PSRAM, забезпечує необхідний апаратний ресурс для розгортання повноцінних моделей глибокого навчання на периферійному пристрої, зберігаючи при цьому стабільність роботи в режимі реального часу.

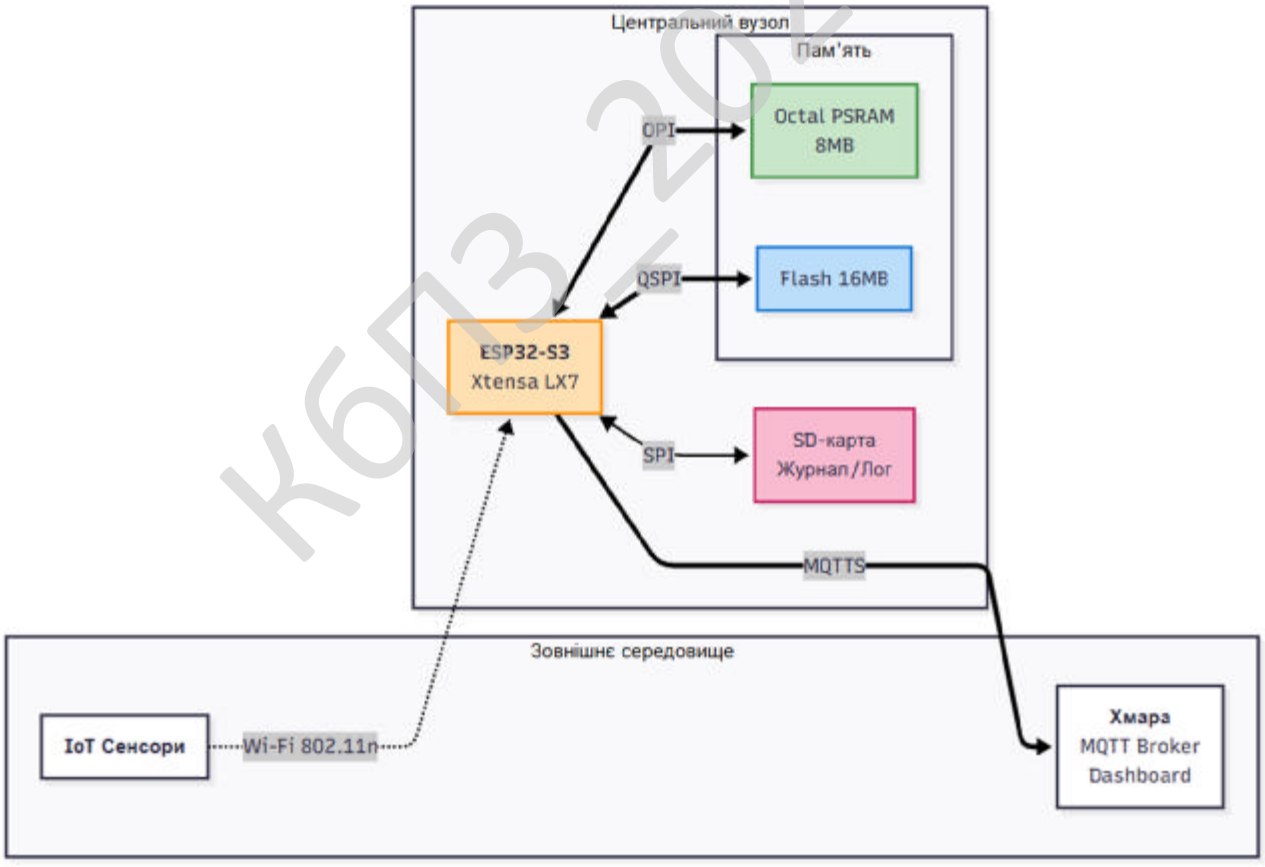


Рисунок 3.1 – Структурна схема апаратно-програмного комплексу

### 3.3 Розробка функціональної схеми

Нафункціональній схемі показано шлях послідовної взаємодії трьох ключових підсистем, кожна з яких відповідає за певний етап життєвого циклу обробки інформації, починаючи від фізичного захоплення сигналу і закінчуючи прийняттям рішень щодо реагування на інцидент з інформаційної безпеки. Взаємодія між функціональними блоками організована з використанням механізмів міжпроцесної комунікації, зокрема черг повідомлень та бінарних семафорів, що гарантує цілісність даних при їх передачі між контекстами виконання різних задач.

Функціонування підсистеми моніторингу та первинної обробки даних розпочинається з процедури ініціалізації радіочастотного тракту мікроконтролера, що передбачає завантаження калібрувальних коефіцієнтів з енергонезалежної пам’яті у регістри PHY. Наступний крок це налаштування драйвера Wi-Fi, який переводиться у режим прослуховування ефіру, відомий як promiscuous mode, шляхом виклику відповідних функцій API ESP-IDF. З метою зниження обчислювального навантаження на центральний процесор, архітектура передбачає застосування апаратного фільтра на рівні MAC-контролера, який налаштовується за допомогою маски фільтрації для відсіювання кадрів даних та службових кадрів, залишаючи для подальшого аналізу виключно кадри керування, що містять критично важливу інформацію про топологію мережі та спроби встановлення з’єднань. Перехоплені пакети, що пройшли первинну фільтрацію та перевірку контрольної суми, передаються до оперативної пам’яті, що дозволяє виконувати перенесення даних без переривання роботи основного обчислювального ядра. Збереження потоку даних організовано у вигляді кільцевого буфера, який розміщається у зовнішній оперативній пам’яті для забезпечення достатньої глибини історії подій. Керування доступом до буфера здійснюється за допомогою двох вказівників, де індекс запису інкрементується драйвером при надходженні нового пакету, а індекс читання оновлюється задачею попередньої обробки після

вилучення даних, причому цілісність вказівників при конкурентному доступі гарантується використанням м'ютексів. Фінальним етапом роботи підсистеми моніторингу є процедура синтаксичного розбору заголовків стандарту IEEE 802.11, під час якої виконується екстракція метаданих, таких як рівень сигналу RSSI, порядковий номер пакету та тип фрейму, з подальшою їх нормалізацією та приведенням до формату цілочисельного вектора ознак int8, що є необхідною умовою для сумісності з вхідним шаром квантованої нейронної мережі.

Передача сформованого вектора ознак запускає роботу підсистеми інтелектуального аналізу, яка виконує функцію ядра для прийняття рішень на основі методів глибокого навчання. На етапі ініціалізації системи виконується завантаження структури нейронної мережі, представленої у форматі FlatBuffer, з файлової системи LittleFS, розміщеної у Flash-пам'яті мікроконтролера, в адресний простір процесора через механізм відображення пам'яті. Критично важливим аспектом функціонування даної підсистеми є менеджмент пам'яті, який реалізується шляхом виділення статичної області, так званої Tensor Arena, у зовнішній пам'яті PSRAM, що дозволяє розмістити всі вхідні, вихідні та проміжні тензори моделі без ризику фрагментації пам'яті в процесі тривалої роботи пристрою. Виконання інференсу нейромережі здійснюється інтерпретатором TensorFlow Lite for Microcontrollers, який послідовно обходить вузли обчислювального графу та викликає відповідні оптимізовані ядра операцій. Для виконання ресурсоємних операцій, таких як одновимірна згортка або повнозв'язні шари, задіюється бібліотека ESP-NN, що використовує набір векторних інструкцій процесора Xtensa LX7 для паралельної обробки даних за схемою SIMD.Данна бібліотекавикористовується для полпшення швидкодії системи. Завершальною операцією циклу аналізу є застосування функції активації до вихідного тензора моделі, що дозволяє перетворити абсолютні значення виходів нейронів у нормований розподіл ймовірностей до одного з відомих класів станів мережі, включаючи нормальну поведінку та різноманітні типи атак.

Отримані результати класифікації передаються до підсистеми керування та реагування, функціонал якої базується на логіці скінченного автомата. Рішення приймаются методом порівняння розрахованої ймовірності атаки із попередньо заданим пороговим значенням, що зберігається у конфігураційному розділі пам’яті, та у випадку перевищення порогу ініціює перехід системи у стан тривоги. Перехід у цей стан активує процедуру реєстрації інциденту, яка полягає у створені детального запису про подію, що включає запис часу, ідентифікатор джерела атаки та рівень впевненості моделі, з подальшим збереженням цього запису на карту пам’яті SD. Паралельно з локальним логуванням виконується формування повідомлення для віддаленого сервера моніторингу, для чого використовується бібліотека cJSON, що створює структурований об’єкт з даними про інцидент. Забезпечення конфіденційності, цілісності та доступності каналу зв’язку реалізується шляхом шифрування вихідного трафіку за протоколом TLS із використанням криптографічних алгоритмів AES та SHA, апаратне прискорення яких забезпечується відповідним модулем мікроконтролера. Транспортний рівень комунікації використовує протоколу MQTT, клієнт якого налаштований на передачу повідомлень з рівнем якості обслуговування QoS 1, що передбачає обов’язкове отримання підтвердження доставки від брокера та повторну відправку пакету у разі виникнення помилок передачі або тайм-ауту [36, 37].

**Графічне представлення функціональної схеми**

Нижче представлено функціональну схему яка відображає ієрархію та внутрішню структуру описаних підсистем.

Ця схема повною мірою показує логіку роботи розробленої програмного забезпечення, демонструючи чіткий поділ на рівні відповідальності та деталізуючи процеси обробки інформації в межах кожного функціонального модуля.

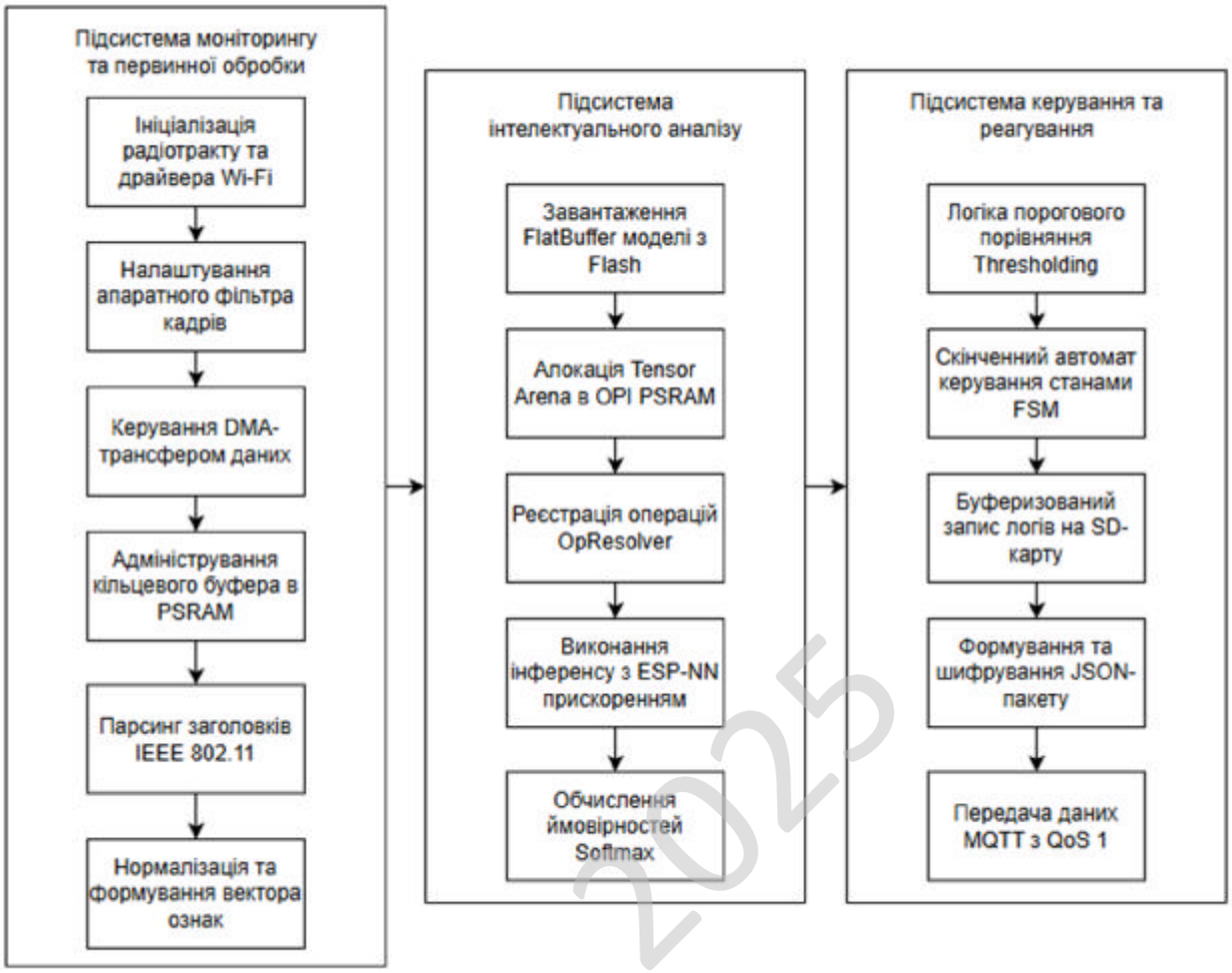


Рисунок 3.2 – Функціональна схема програмного забезпечення



### 3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання магістерської роботи, наведена на рисунку 3.3. При детальному її розгляді можна побачити, як саме проходить взаємодія у розробленій системі.

Використовується модель проектування, що представляє графічне відображення «потоків» даних у ній. Діаграма використовується для візуалізації процесів обробки даних. Для розробника вважається нормальним спочатку креслити таку діаграму, завдяки чому буде показано взаємодію системи. Вона в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних для того щоб показати систему, що розробляється.

Життєвий цикл обробки інформації розпочинається з процесу «Ініціалізація та конфігурація». На цьому етапі відбувається завантаження драйверів, перевірка цілісності файлової системи та зчитування всіх налаштувань. Успішне завершення ініціалізації передає потік керування до процесу «Моніторинг ефіру», який переводить Wi-Fi модуль у режим прослуховування для перехоплення кадрів стандарту IEEE 802.11.

Процесякий запобігає втраті даних при пікових навантаженнях називається «Керування кільцевим буфером». Захоплені пакети асинхронно записуються у виділену ділянку оперативної пам'яті, що дозволяє розв'язати задачу синхронізації між високошвидкісним потоком вхідних даних та алгоритмами аналізу.

Наступним етапом є вибірка даних з буфера та їх передача до процесу «Парсинг та Нормалізація». Тут відбувається синтаксичний розбір заголовків пакетів, виділення ключових метаданих (RSSI, тип фрейму, часові мітки) та їх перетворення у нормалізований числовий вектор (формат int8), придатний для обробки нейронною мережею. Сформований вектор надходить до процесу «Нейромережевий Інференс», який використовує завантажену модель TensorFlow Lite для обчислення ймовірності належності трафіку до класу кібератак.

Результат інференсу (Score) передається до процесу «Перевірка порогу». Цей процес є точкою розгалуження потоку даних:

У випадку, якщо ймовірність атаки не перевищує заданий поріг, система повертається до стану моніторингу.

При виявленні аномалії активується ланцюжок процесів реагування. Спочатку дані передаються до процесу «Шифрування даних» (алгоритм AES), після чого захищений пакет паралельно спрямовується до процесів «Запис на SD» для локального аудиту та «Відправка на сервер» через протокол MQTT.

Окремою складовою діаграми є процес «Обробка помилок», який має зворотні зв'язки з усіма критичними вузлами системи. У разі виникнення помилки (переповнення буфера, збій інференсу, втрата зв'язку) цей процес ініціює процедуру відновлення або повного перезавантаження системи.

На рисунку 3.3 зображено деталізовану діаграму процесів, що відображає логіку функціонування розробленого програмного забезпечення.

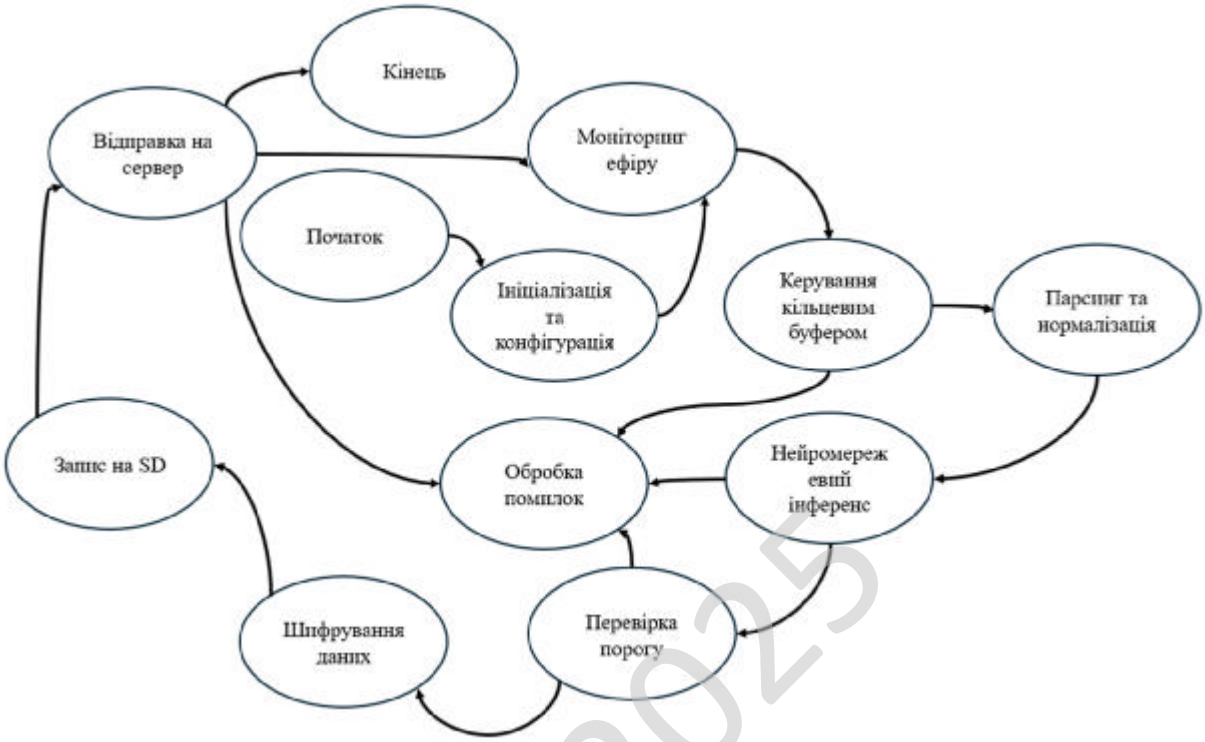


Рисунок 3.3 – Діаграма взаємодії процесів

# 4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

## 4.1 Розробка блок-схем та опис алгоритмів функціонування системи

Розроблене програмне забезпечення апаратно-програмного комплексу функціонує за циклічним принципом під керуванням операційної системи реального часу FreeRTOS. Алгоритмічна структура головного циклу обробки даних (Main Processing Loop) побудована таким чином, щоб забезпечити детермінований час реакції на події безпеки, мінімізувати затримки при обробці мережевих пакетів та гарантувати надійність функціонування в умовах безперервної експлуатації. Блок-схема алгоритму, що налічує 19 функціональних блоків, відображає послідовність операцій від моменту ініціалізації до завершення роботи.

Процес виконання алгоритму розпочинається з термінального блоку «Початок», який відповідає моменту подачі живлення на мікроконтролер ESP32-S3 та запуску завантажувача. Першим етапом є процедура «Ініціалізація периферії та Wi-Fi». На цьому кроці виконується налаштування апаратних драйверів (HAL), перевірка цілісності зовнішньої оперативної пам’яті PSRAM, монтування файлової системи Flash-пам’яті та переведення радіомодуля у режим прослуховування (Promiscuous Mode). Успішне завершення ініціалізації переводить систему в стан очікування вхідних даних.

Центральним елементом циклу є умовний блок «Пакет отримано?». Алгоритм перевіряє стан черги повідомлень FreeRTOS на наявність нових дескрипторів захоплених кадрів. У випадку негативного результату (відсутність мережевої активності) потік керування оминає блоки обробки та переходить безпосередньо до блоку діагностики, що дозволяє економити обчислювальні ресурси. У разі

позитивного результату виконується перехід до «Зчитування даних з кільцевого буфера», де відбувається копіювання «сирого» пакету з області DMA у локальну пам’ять задачі.

Наступним етапом є «Парсинг заголовків та нормалізація». На цьому кроці виконується синтаксичний розбір структури кадру IEEE 802.11, виділення метаданих (RSSI, Sequence Control) та їх перетворення у нормалізований вектор ознак формату int8. Підготовлені дані передаються у «Підпрограму інференсу TFLite», яка реалізована як окрема функціональна одиниця (Predefined Process). Результатом роботи підпрограми є тензор ймовірностей, який фіксується у блоці «Отримання результату класифікації».

Дуже важливим є розгалуження «Атаку виявлено (Score > Threshold)?». Алгоритм порівнює отриману ймовірність із заданим пороговим значенням. Якщо умова не виконується (атаку не виявлено), система переходить до блоку діагностики, ігноруючи процедури реагування. Якщо умову істинно (атаку підтверджено), виконується лінійна послідовність дій з реагування:

1. Формування пакету тривоги (JSON) це серіалізація даних про інцидент у текстовий формат.

2. Шифрування даних (AES-256) являє собою криптографічний захист корисного навантаження для забезпечення конфіденційності.

3. Відправка повідомлення MQTT це передача зашифрованих даних на сервер моніторингу.

4. Оновлення системного таймера являє собою синхронізацію часу для точної фіксації моменту події.

5. Запис події на SD-карту це збереження інформації на локальний енергонезалежний носій для подальшого аудиту.

Незалежно від результату аналізу (наявність або відсутність атаки), потоки керування сходяться у точці перевірки стану системи — блок «Діагностика: помилки системи?». Тут аналізуються прапори переповнення буферів, помилки шини SPI або збої Wi-Fi стека. За відсутності помилок алгоритм переходить до

перевірки умови завершення. У разі виявлення збоїв активується «Підпрограма обробки помилок», після чого виконується «Скидання буферів та Watchdog» для відновлення стабільності та «Вивід статусу помилки в лог».

Завершальним етапом ітерації є перевірка умови «Сигнал завершення роботи?». Якщо отримано команду на зупинку (наприклад, для оновлення прошивки), алгоритм завершується у блоці «Кінець». В іншому випадку здійснюється повернення до етапу перевірки наявності нових пакетів.

**Графічне представлення алгоритму**

На рисунку 4.1 наведено блок-схему основного алгоритму функціонування системи.

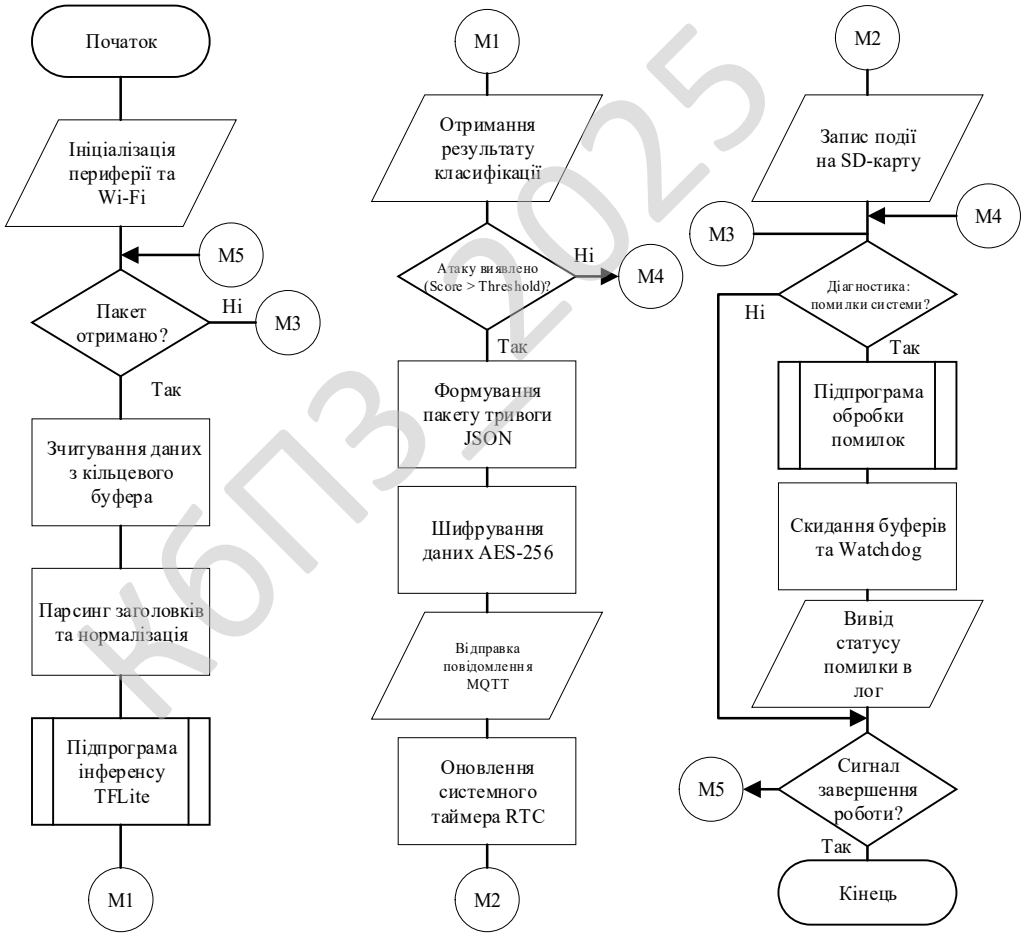


Рисунок 4.1 – Блок-схема основної програми

**Алгоритм підпрограми нейромережевого інференсу**

Виконання підпрограми розпочинається з отримання вхідного аргументу у вигляді вказівника на область пам’яті, що містить буферизований пакет даних. Першим етапом обробки є процедура нормалізації та квантування отриманих значень. Оскільки вхідний шар нейронної мережі сконфігуровано для роботи з цілочисельними даними фіксованої розмірності, здійснюється перетворення значень з плаваючою комою у формат int8. Дана операція виконується шляхом застосування афінного перетворення з використанням коефіцієнтів масштабування (scale) та зміщення нуля (zero point), визначених на етапі тренування моделі. Підготовлений масив даних копіюється безпосередньо у вхідний тензор, розміщений у статично виділеній області пам’яті, що забезпечує коректне розміщення даних у пам’яті PSRAM перед початком обчислень.

Наступним кроком ініціюється основний обчислювальний процес шляхом виклику методу Interpreter::Invoke() який знаходится в бібліотеці TensorFlow Lite for Microcontrollers. Ця операція є синхронною та блокуючою, під час якої виконується послідовний обхід вузлів обчислювального графу моделі. Інтерпретатор задіює оптимізовані ядра задля математичних операцій, використовуючи векторні інструкції цифрового сигнального процесора для прискорення виконання згорток та матричних множень. По завершенню обчислень здійснюється перевірка коду повернення функції Invoke на відповідність значенню kTfLiteOk. У випадку виявлення помилки виконання підпрограми переривається та повертає відповідний код помилки викликаючому процесу. При успішному завершенні інференсу виконується етап постібробки результатів, що полягає у застосуванні функції активації Softmax до вихідного тензора моделі. Сирі значення виходів нейронів трансформуються у нормований розподіл ймовірностей у діапазоні від 0.0 до 1.0. Розраховане значення ймовірності повертається в основну програму для порівняння з пороговим значенням. Графічне представлення алгоритму підпрограми інференсу наведено на рисунку 4.2.

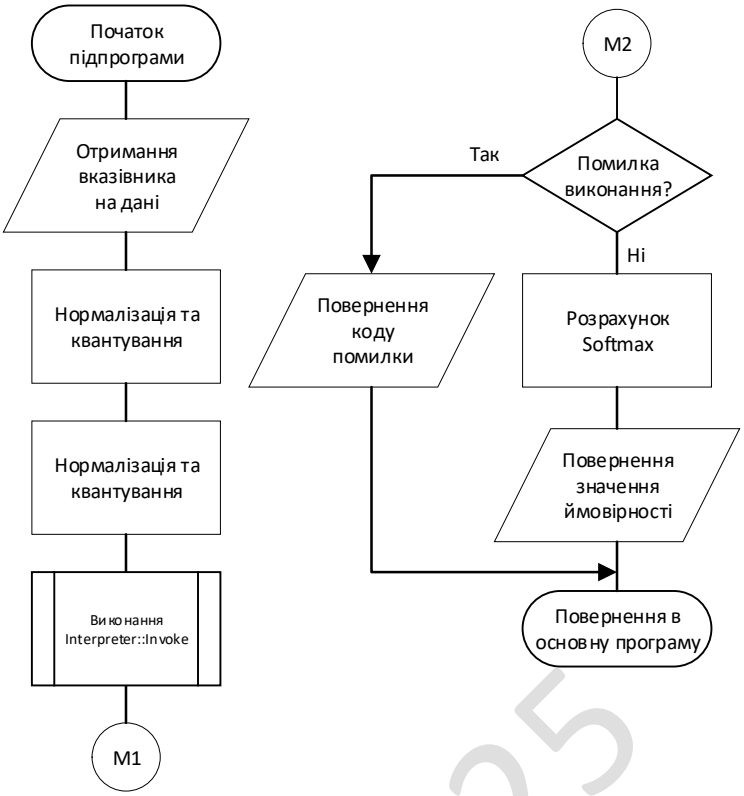


Рисунок 4.2 - Підпрограма нейромережевого інференсу

**Алгоритм підпрограми обробки помилок**

Робота алгоритму обробки виключних ситуацій розпочинається отриманням коду помилки від основного циклу програми або драйверів. Першою дією алгоритму є примусова зупинка планувальника задач операційної системи. Такий підхід забезпечує монопольний доступ до ресурсів процесора та запобігає перемиканню контексту під час виконання критичних операцій діагностики, гарантуючи збереження цілісності стану системи на момент виникнення збою.

Далі виконується логічне розгалуження на основі аналізу типу отриманої помилки. Здійснюється перевірка умови критичності збою. У разі ідентифікації помилки як критичної (порушення цілісності стеку, помилки доступу до пам’яті), алгоритм переходить до процедури аварійного завершення роботи. Виконується формування повного дампу стану регістрів процесора та пам’яті та його запис у енергонезалежну пам’ять. Після збереження діагностичної інформації здійснюється активація та налаштування сторожового таймера, що призводить до примусового перезавантаження мікроконтролера.

Якщо помилка класифікується як некритична, то розпочинається відновлення. Формується текстовий запис про інцидент з міткою часу, який зберігається на карті пам’яті SD через інтерфейс SPI. Наступним кроком здійснюється примусове очищення програмних буферів даних та скидання прапорів помилок драйверів. Завершується алгоритм відновленням роботи планувальника задач викликом xTaskResumeAll() та поверненням керування в основну програму для продовження штатного режиму функціонування. Блок-схема алгоритму обробки помилок наведена на рисунку 4.3.

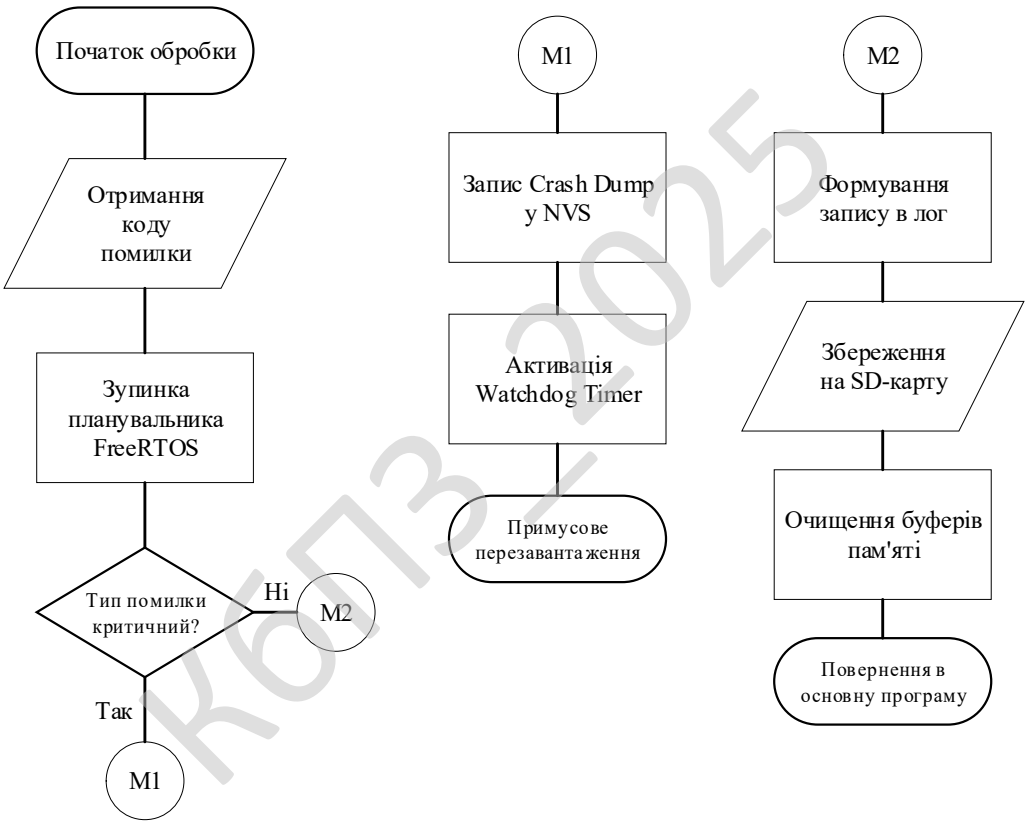


Рисунок 4.3 - Алгоритм підпрограми обробки помилок



### 4.2 Захист розробленого програмного забезпечення

Безпека розробленої системи та захист програмного забезпечення базується на концепції апаратного кореня довіри,яка запроваджена на рівні архітектури мікроконтролера ESP32-S3. Ключовим компонентом даного механізму є блок

одноразово програмованої пам’яті eFuse, який складається з масиву фізичних перемикачів, стан яких визначає конфігурацію безпеки пристрою. У даній області пам’яті зберігаються криптографічні ключі та конфігураційні біти безпеки, внесення змін або програмне зчитування вмісту яких після активації захисту (стає фізично неможливим.

Забезпечення цілісності виконавчого коду та захист від підміни завантажувача реалізується через механізм Secure Boot V2. Процес верифікації якого базується на використанні асиметричного алгоритму RSA з довжиною ключа 3072 біти. Цей алгоритм перевірки цілісності можна описати наступним чином. Нехай F - бінарний образ прошивки, а S - відповідний цифровий підпис. На першому етапі завантажувач виконує обчислення хешу прошивки за допомогою такої хеш-функції:

$$H = SHA256(F), \tag{4.1}$$

де H - результат хешування довжиною 256 біт. Далі здійснюється процедура верифікації підпису S із використанням відкритого ключа $K_{pub}$, хеш якого заздалегідь записаний у блок eFuse:

$$V = RSA\_Verify(S, K_{pub}) \tag{4.2}$$

Умовою успішного завантаження та передачі керування прикладній програмі є виконання рівності: H == V. У разі виявлення невідповідності завантаження припиняється, що унеможливює виконання модифікованого коду.

Конфіденційність даних, зокрема параметрів нейромережевої моделі, забезпечується через шифрування зовнішньої пам’яті. В основі захисту лежить алгоритм AES-256, що функціонує в режимі XTS. Унікальний ключ шифрування $K_{flash}$ генерується внутрішнім апаратним генератором випадкових чисел безпосередньо всередині мікроконтролера під час першого завантаження і ніколи не передається за межі чіпа. Процес перетворення відкритого тексту P у зашифрований текст C враховує фізичну адресу блоку даних у пам’яті, що описується функціональною залежністю:

$$C = AES_{XTS}(P, K_{flash}, Addr) \tag{4.3}$$

Даний підхід забезпечує унікальність шифрованого тексту для однакових фрагментів даних, розташованих за різними адресами, та унеможливлює копіювання прошивки на інші пристрої.

Захист каналу передачі даних та команд керування реалізовано за допомогою протоколу MQTTS, що передбачає інкапсуляцію трафіку в захищений тунель TLS версії 1.3. Щоб інформація була конфіденційною використовується алгоритми симетричного шифрування AES-128/256-GCM, а автентифікація віддаленого сервера - через перевірку ланцюжка сертифікатів стандарту X.509. Для забезпечення високої швидкодії та мінімізації затримок у роботі, операції модульної експоненти та AES виконуються із залученням вбудованого криптографічного акселератора. Це дозволяє реалізувати криптографічні перетворення на апаратному рівні, забезпечуючи стійкість системи до мережевих атак та високу пропускну здатність каналів зв’язку.

# 5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено інтерфейс адміністративної панелі, розробленої у результаті виконання магістерської роботи.

Розроблена адміністративна панель складається з наступних функціональних блоків:

— Головне навігаційне меню: файл (налаштування підключення); Моніторинг (спостереження в реальному часі); аналітика (статистика загроз); безпека (керування вузлами).

— Графічна панель візуалізації даних: відображення розподілу типів атак та часової динаміки трафіку у вигляді інтерактивних діаграм та гістограм.

— Розділ конфігурації з’єднання: блок налаштувань для встановлення захищеного зв'язку з MQTT-брокером та вибору мережевого інтерфейсу для прослуховування ефіру.

— Інформаційна панель стану системи: відображення статусу підключення до шлюзу безпеки, кількості виявлених активних загроз та ізольованих пристроїв.

— Вікно результатів інференсу: область виведення ймовірнісних оцінок нейромережі, деталізованих логів аномальної активності та структурованих JSON-пакетів.

— Елементи контекстного керування: інтерактивні кнопки, що дозволяють розпочати поглиблений аналіз або надсилати команди ізоляції/відновлення для конкретного пристрою.

— Деревовидне представлення топології IoT-мережі: візуалізація ієрархії підключених сенсорів та пристроїв із відображенням їхніх ідентифікаторів (MAC/IP) та актуального статусу захищеності.

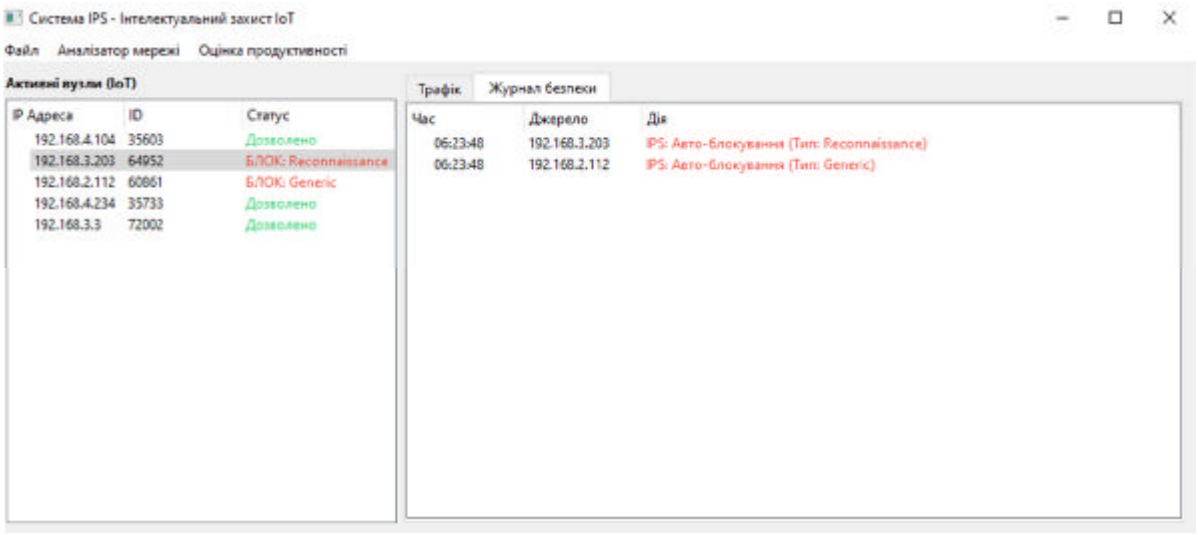


Рисунок 5.1 – Головне вікно розробленого ПЗ

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

**Проводилось тестування форматом білої скриньки** засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

– Гарантується перевірка всіх незалежних маршрутів програми.

– Знаходяться гілки True, False для всіх логічних рішень.

– Виконуються всі цикли (у межах їхніх кордонів та діапазонів).

– Аналізується правильність внутрішніх структур даних.

Недоліки тестування "білої скриньки":

– Кількість незалежних маршрутів може бути дуже велика.

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

Обрано умови розповсюдження – commercial software.

Програмне забезпечення, створене комерційною організацією з метою отримання прибутку від його використання іншими, наприклад, шляхом продажу копій.

Найважливішою особливістю комерційний програмних продуктів є підтримка великих компаній, прямо зацікавлених у поширенні програм. Багато організацій

надають виключно платну підтримку своїх продуктів, такий підхід, як правило, використовують організації надають відкриті вихідні коди. Для продуктів, що розповсюджуються на комерційній основі діють зазвичай безкоштовні служби підтримки, покликані збільшити рівень довіри у клієнтів і потенційних покупців.

Далеко не завжди, але як правило часові рамки для важливих змін в комерційних продуктах значно менше, ніж у некомерційних проектів. Це пов’язано з тим, що над комерційним продуктом працюють цілі групи розробників і ця робота є їх основним заняттям. Розробникам-початківцям як правило доводиться шукати додаткові способи заробітку, і це збільшує час, що витрачається на доповнення і зміни програм. Так як основним рушійним фактором створення комерційного ПЗ є одержання прибутку, то комерційні програмні продукти першими заповнюють вільні ніші та пропонують варіанти вирішення завдань відразу по мірі виявлення вакууму в будь-якому секторі ринку.

Окремий вид комерційних програм, коли їх розробка оплачується безпосередньо замовником. Такі програми найчастіше позбавлені всіх переваг комерційних продуктів, оскільки мають обмежений бюджет, але більш адаптовані до вимог замовника, ніж аналоги.

# 6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для апаратно-програмного комплексу виявлення атак на рівні IoT-пристроїв з використанням легковажих нейромереж.

Метою розробки є дослідження та програмна реалізація інтелектуальної системи захисту IoT-інфраструктури на основі методів машинного навчання на граничних пристроях.

Об’єктом дослідження є процес функціонування та обміну даними в мережах інтернету речей в умовах кібернетичного впливу та ресурсних обмежень апаратної платформи.

Предметом дослідження є методи та апаратно-програмні засоби нейромережевого виявлення аномалій, оптимізовані для виконання на мікроконтролерах.

Методи дослідження базуються на методах системного аналізу, теорії штучних нейронних мереж, методах математичної статистики та методах розробки вбудованого програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

— Удосконалено метод виявлення мережевих атак на рівні кінцевих пристроїв шляхом адаптації архітектури нейронної мережі до специфіки векторних інструкцій мікроконтролера ESP32-S3, що дозволило значно зменшити час обробки трафіку при збереженні високої точності класифікації.

— Набув подальшого розвитку підхід до побудови автономних шлюзів безпеки для мереж Інтернету речей, який, на відміну від існуючих хмарних рішень, забезпечує локальну ізоляцію скомпрометованих вузлів та прийняття рішень

безпосередньо на периферії мережі без необхідності звернення до центрального сервера.

# 7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ІТ-ПРОЄКТУ

## Визначення цільової аудиторії кінцевого готового продукту

Визначення цільової аудиторії розробленого апаратно-програмного комплексу базується на аналізі потреб ринку в автономних та бюджетних засобах кіберзахисту. Оскільки система використовує технології граничних обчислень на базі мікроконтролера, вона орієнтована на користувачів, для яких критичними є швидка реакція на загрози та незалежність від хмарних сервісів.

Основним споживачем продукту є промисловий сектор, де підприємства потребують захисту мереж від несанкціонованого втручання без передачі даних за межі локальної мернежі. Локальний аналіз трафіку нейромережею дозволяє оперативно ізолювати загрози, що важливо для операторів критичної інфраструктури, які мають підвищені вимоги до безпеки та конфіденційності.

В сегменті споживчої електроніки та систем «Розумного будинку» розроблений комплекс виступає як захисний шлюз. Він забезпечує безпеку для великої кількості пристроїв різних виробників, які часто не мають вбудованих засобів захисту та рідко отримують оновлення безпеки. Компаніїякі інтегрують такі системи можуть використовувати даний продукт для підвищення загального рівня захищеності мереж своїх клієнтів.

Також система приваблива для компаній, що надають послуги з моніторингу кібербезпеки. Завдяки підтримці протоколу MQTTS та наявності адміністративної панелі на Python, комплекс легко стає частиною великих центрів моніторингу як доступний периферійний датчик виявлення аномалій. Комерційний успіх проєкту забезпечується поєднанням низької вартості апаратної частини та високої ефективності інтелектуального аналізу даних.

## 7.2 Оцінка привабливості шляхом застосування методів експертних оцінок

Щоб перевірити, наскільки розроблений комплекс буде затребуваним на реальному ринку, я провів аналіз його характеристик за допомогою методу експертних оцінок. Це дозволило поглянути на систему не просто як на наукову розробку, а як на потенційний продукт, оцінюючи його очима фахівців з кібербезпеки та автоматизації. В основу оцінювання лягли такі критерії, як технічна новизна, надійність захисту, простота впровадження та, що дуже важливо для комерційного сектору, економічна вигідність.

Найвищі бали експерти поставили за використання концепції Edge AI, оскільки запуск нейронної мережі безпосередньо на мікроконтролері ESP32-S3 дозволяє системі працювати автономно. Фахівці відзначили, що адаптація алгоритмів під векторні інструкції архітектури Xtensa LX7 робить обробку трафіку дуже швидкою, що критично для виявлення атак у реальному часі. Крім того, професійну зацікавленість викликав той факт, що в системі з самого початку закладено апаратні функції захисту, такі як Secure Boot V2 та шифрування пам’яті за стандартом AES-256 XTS, що робить розробку стійкою до фізичного втручання та підміни коду.

Окремо обговорювалася практична цінність механізму дистанційної ізоляції вузлів через MQTT-команди. Експерти зійшлися на думці, що можливість миттєво заблокувати зловмисника через зручну адмін-панель на Python є вагомою перевагою для системного адміністратора. При цьому використання протоколу MQTTS із захистом TLS 1.3 знімає питання щодо безпеки самих керуючих команд.

З точки зору економіки, мій проєкт виглядає виграшно, бо використання недорогої платформи ESP32-S3 дозволяє створити потужний захисний шлюз за ціною, значно нижчою, ніж у класичних промислових IDS. Загальний підсумок експертного аналізу підтвердив, що продукт вдало заповнює вільну нішу на ринку

безпеки IoT, поєднуючи в собі інтелектуальні можливості великих систем із компактністю та низькою ціною вбудованих рішень.

**Вибір методу оцінки вартості ПЗ**

Щоб вибрати метож оцінки вартості ПЗ необхідно врахувати не лише витрати на написання коду, а й на апаратну реалізацію периферійних обчислень. Для обґрунтування інвестиційної доцільності впровадження системи обрано метод повної вартості володіння. Даний підхід дозволяє комплексно проаналізувати всі прямі та непрямі витрати, що виникають протягом повного життєвого циклу експлуатації ситеми - від розробки архітектури TinyML до етапу технічної підтримки в промислових умовах.

Використання цього методу для цього проєкту зумовлене можливістю деталізації наступних видатків:

— Капітальні витрати: придбання мікроконтролерів ESP32-S3, розробка спеціалізованих друкованих плат, а також часові витрати на навчання нейромережевих моделей у середовищі TensorFlow Lite.

— Операційні витрати: мінімальне енергоспоживання архітектури Xtensa LX7 (що критично для автономних вузлів), витрати на адміністрування MQTT-брокера та забезпечення безпечного оновлення прошивки за технологією OTA.

Застосування даної методики дозволяє чітко знайти потенційні зони оптимізації бюджету. Зокрема, перенесення процесу обробки трафіку безпосередньо сам мікроконтролер радикально знижує витрати на мережу та оренду хмарних серверних потужностей, які зазвичай необхідні для аналізу великих масивів даних.

Інтеграція методу TCO з аналізом вигід забезпечує зіставлення вкладених ресурсів із отриманими результатами: підвищенням рівня стійкості локальної мережі до DDoS-атак та зменшенням часу реакції на інциденти. Такий підхід робить оцінку вартості прозорою для управлінських структур, підкреслюючи

перевагу використання бюджетної, але потужної платформи ESP32-S3 над дорогими промисловими аналогами. У результаті, обраний метод дозволяє довести, що економічна ефективність системи досягається не лише за рахунок низької ціни компонентів, а й завдяки автономності та енергоефективності програмних рішень.

## 7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості

Перенесення інференсу нейронної мережі безпосередньо на мікроконтролер дозволяє замінити дорогі серверні рішення автономними вузлами з низьким енергоспоживанням. Обґрунтування привабливості проєкту проведено на прикладі локальної мережі промислового об’єкта з 50 точками моніторингу трафіку.

Для розрахунку використано наступні показники:

— Зниження витрат на хмарні сервіси: використання локального аналізу замість API-запитів до Cloud AI дозволяє заощадити близько 4 500 грн на місяць для одного вузла.

— Економія мережевого ресурсу: фільтрація 98% надлишкового трафіку на рівні Edge мінімізує витрати на пропускну здатність каналів зв’язку.

— Енергоефективність: сумарна потужність 50 модулів ESP32-S3 не перевищує 25 Вт, що в десятки разів менше за споживання типового сервера аналізу даних (від 250 Вт), забезпечуючи річну економію електроенергії обсягом понад 15 000 грн.

Фінансові результати впровадження системи демонструють високу рентабельність, а саме після вирахування витрат на адміністрування MQTT-інфраструктури становить 420 000 грн для досліджуваної мережі завдяки низькій собівартості апаратної частини інвестиції повертаються протягом 3,5 місяців експлуатації; коефіцієнт ROI складає 285%, що підтверджує високу фінансову стійкість проєкту.

Порівняльні характеристики систем наведено у таблиці 7.1.

Таблиця 7.1 - Порівняння витрат та характеристик систем захисту

| Показник | Хмарна IDS (традиційна) | Edge AI шлюз (ESP32-S3) |
|---|---|---|
| Вартість апаратної точки | 12 000 грн | 950 грн |
| Щомісячна абонентська плата | 3 500 грн | 0 грн |
| Затримка аналізу | 150–400 мс | 15–30 мс |
| Енергоспоживання | 1800 кВт·год | 12 кВт·год |
| Ризик компрометації даних | Середній | Мінімальний |

Поряд із прямими грошовими вигодами, впровадження комплекс забезпечує і нефінансові ефекти. Локальна обробка даних нейромережею зводить затримку на реакцію до інциденту до мінімуму. Відсутність передачі конфіденційних пакетів за межі локального контуру автоматично підвищує рівень захисту від перехоплення даних.

### 7.5 Пропозиція алгоритму просування проєкту розробки ПЗ

Оюрана стратегія виходу на ринок базується на демонстрації технічної переваги TinyML-алгоритмів над хмарними аналогами. Просування розділено на три логічні етапи, що дозволяє поступово нарощувати довіру професійної спільноти та замовників.

Першочерговим кроком є запуск пілотних майданчиків (MVP) у реальних умовах. Візуалізація того, як мікроконтролер ESP32-S3 самостійно ідентифікує атаку та миттєво ізолює вузол, є ключовим фактором переконання технічних директорів (CTO). Прямий показ результатів роботи нейромережі на периферії виключає скептицизм щодо обчислювальних можливостей вбудованих систем.

На другому етапі акцент зміщується на цифрову присутність у професійних мережах. Публікація оптимізованого коду на GitHub та опис архітектурних рішень

у LinkedIn дозволяють залучити увагу системних адміністраторів та розробників IoT-рішень. Участь у профільних конференціях (наприклад, з питань кібербезпеки критичної інфраструктури) закріплює статус розробки як надійного вітчизняного продукту.

Заключна стадія передбачає налагодження B2B-партнерств із постачальниками обладнання для «розумних будинків» та системними інтеграторами Industrial IoT. Створення готових модулів, що легко інтегруються в наявні шафи керування, дозволяє реалізувати модель продажу «під ключ». Такий підхід мінімізує витрати на рекламу, фокусуючись на прямому вирішенні проблем безпеки конкретних підприємств.

### 7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ

Пріоритетним каналом збуту буде співпраця з компаніями, що займаються автоматизацією промислових об’єктів, де розроблена система впроваджується як готовий автономний модуль захисту. Це дозволяє використовувати наявні мережі збуту партнерів для швидкого охоплення ринку без значних витрат на власну логістику та маркетинг. Окремим шляхом реалізації є ліцензування програмного стека TinyML для сторонніх виробників електроніки, що дозволяє вбудовувати розроблені алгоритми безпосередньо в їхні апаратні платформи.

Оптимізація впровадження передбачає перехід до гібридної сервісної моделі, де одноразовий продаж обладнання поєднується з підпискою на оновлення нейромережевих моделей. Дистанційна доставка прошивок через захищений механізм OTA гарантує актуальність бази загроз на периферійних вузлах без необхідності фізичного доступу спеціаліста до пристроїв. Цифрова присутність у репозиторіях професійного спрямування та кейси в мережі LinkedIn допомагають залучати технічних фахівців і скорочувати цикл ухвалення рішення про закупівлю. Такий підхід забезпечує сталий попит і стабільний розвиток проєкту завдяки низькій вартості масштабування та високій автономності розроблених рішень.

### 7.7 Визначення ключових факторів успіху конкретного проєкту

Ключовий успіх розробленого комплексу полягає у поєднанні інтелектуальної автономності Edge AI та апаратної стійкості мікроконтролера ESP32-S3. Завдяки використанню векторних інструкцій архітектури Xtensa LX7 обробка мережевого трафіку відбувається практично миттєво, що дозволяє ідентифікувати загрози в реальному часі без звернення до хмари.

Надійність розробки підсилюється вбудованими функціями Secure Boot V2 та шифруванням пам'яті AES-256 XTS, які запобігають фізичному втручанню, тоді як протокол MQTTS із захистом TLS 1.3 гарантує конфіденційність керуючих команд. Висока економічна вигідність платформи у поєднанні з можливістю автономної ізоляції атак робить систему конкурентоспроможною, заповнюючи вільну нішу доступних інтелектуальних засобів захисту для IoT.

# 8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

## 8.1 Вступ

Регулювання трудових відносин у межах вітчизняного ІТ-сектору нерозривно пов’язане з виконанням вимог Закону України «Про охорону праці», що зобов’язує кожне підприємство до впровадження комплексних захисних заходів. Процес управління безпекою персоналу реалізується через систематизацію інструкцій за професіями, розробку внутрішніх положень, наказів та ведення журналів реєстрації інструктажів. Роботодавець забезпечує створення профільного відділу, функціонування якого повністю відповідає державній політиці та типовим нормативам центральних органів виконавчої влади. Ігнорування встановлених законодавчих правил тягне за собою адміністративну відповідальність у вигляді штрафів, а виникнення випадків травмування працівників призводить до притягнення відповідальних осіб до кримінальної відповідальності.

Основоположним інструментом деталізації безпекових норм є Наказ Міністерства соціальної політики № 207, що регламентує захист здоров’я під час роботи з екранними пристроями. Професійна діяльність інженерів-програмістів супроводжується специфічним негативним впливом на органи зору, високою розумовою напругою та інтенсивним нервово-емоційним навантаженням. Постійна взаємодія з периферійними пристроями введення створює значне статичне напруження м'язів та суглобів рук. Додаткову загрозу в процесі експлуатації апаратної частини обчислювальних систем становлять високочастотні електромагнітні випромінювання та погіршення складу повітря через виділення шкідливих газів.

Формування безпечних умов праці базується на положеннях ДСанПіН 3.3.2-007-98 та вимогах НПАОП 0.00-7.15-18, що передбачають ретельний контроль параметрів мікроклімату, освітленості та ефективності вентиляції приміщень.

Системний аналіз цих факторів дозволяє вчасно ідентифікувати потенційні причини виникнення професійних захворювань. Своєчасне визначення чинників шкідливого впливу є необхідним підґрунтям для розробки дієвої стратегії захисту організму розробника та підтримки його високої працездатності протягом усього життєвого циклу проєкту.

### 8.2 Шкідливі і небезпечні фактори при роботі з комп'ютером

Експлуатація апаратної платформи та супутнього обладнання створює умови для виникнення низки потенційно небезпечних факторів, серед яких пріоритетним визначено ризик ураження електричним струмом у разі порушення цілісності ізоляції блоків живлення. Оскільки проектування архітектури нейромережевих моделей та подальший аналіз мережевого трафіку супроводжуються інтенсивним зоровим навантаженням, особливої ваги набуває організація належної освітленості робочої зони. Робоче середовище інженера-розробника TinyML-рішень повинно відповідати встановленим параметрам мікроклімату, регулювання яких здійснюється згідно з нормами ДСанПіН 3.3.2.007-98 для забезпечення стабільної працездатності організму.

Процес розробки та валідації автономних шлюзів безпеки характеризується впливом специфічних високочастотних електромагнітних випромінювань, що генеруються радіомодулями Espressif у режимах активного сканування ефіру та передачі телеметрії. Поряд із технологічними ризиками, до яких належить імовірність виникнення пожежі через перегрів електронних компонентів або перевантаження силових контурів, існують значні психофізіологічні загрози. Інтелектуальне навантаження зумовлене необхідністю постійного розрізнення аномалій у складних часових рядах даних, що потребує високого ступеня концентрації уваги при мінімальних часових затримках. Невідповідність ергономічних показників робочого місця актуальним стандартам призводить до

виникнення статичних перевантажень кістково-м’язового апарату, зокрема в області суглобів кистей рук під час інтенсивного введення коду.

Діяльність фахівця з комп'ютерної інженерії класифікується як психічна форма праці з високим ступенем напруженості, що пов’язано з безперервним відстеженням динаміки інференсу нейромережі та читанням технічної документації. Монотонність виконання окремих операцій у поєднанні з акустичним шумом від систем охолодження серверного обладнання створює умови для накопичення нервово-емоційної втоми. Будь-яка активність у межах реалізації даного проєкту супроводжується мобілізацією вищих психічних функцій, оскільки ціна помилки при налаштуванні порогів чутливості нейродетектора може призвести до компрометації всієї IoT-інфраструктури. Таким чином, сукупний вплив кількох десятків шкідливих чинників вимагає впровадження дієвих заходів захисту для нейтралізації професійних ризиків.

### 8.3 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Оцінювання гігієнічної адекватності робочого простору базується на аналізі геометричних характеристик приміщення, де здійснюється експлуатація обчислювальної техніки та розробка нейромережевих моделей. Досліджувана робоча зона має ширину 5 метрів, довжину 6,2 метра та висоту 3,4 метра, що сумарно забезпечує загальну площу 31 м² та корисний об’єм 105,4 м³. Враховуючи одночасне перебування у приміщенні двох фахівців, на кожного працюючого припадає 15,5 м² площі та 52,7 м³ об’єму. Такі показники суттєво перевищують мінімальні пороги, встановлені ДСанПіН 3.3.2-007-98 та Наказом Міністерства соціальної політики № 207, що гарантує відсутність ефекту обмеженості простору та сприяє нормальному газообміну в процесі трудової діяльності.

Термічний режим середовища формується під впливом внутрішніх тепловиділень від апаратної частини ESP32-S3, системних блоків ЕОМ, освітлювальних приладів та безпосередньо працюючого персоналу. Процес

розробки TinyML-рішень класифікується як робота категорії Іа, що характеризується низькими енерговитратами організму на рівні до 120 ккал/год. Підтримка оптимальних параметрів мікроклімату реалізується через функціонування систем кондиціонування у теплий період та штучного опалення у холодну пору року. Фактична температура повітря у діапазоні 22–25 °С при відносній вологості 40–65 % повністю корелює з нормативними вимогами, забезпечуючи стабільну когнітивну активність розробника. Мінімізація запиленості досягається шляхом регламентованого провітрювання та проведення вологого прибирання робочих поверхонь.

Акустичний фон у приміщенні створюється переважно аеродинамічним шумом систем охолодження комп'ютерної техніки та періодичною роботою принтера. Сумарний рівень звукового тиску не виходить за межі допустимих значень, що дозволяє уникнути нервового виснаження та підтримувати високу концентрацію уваги при аналізі великих масивів даних. Окремим критичним чинником виступає світлотехнічне забезпечення робочої зони, що регулюється нормами ДБН В.2.5-28:2018. Робота програміста ідентифікована як діяльність V розряду зорової роботи підрозряду В, що вимагає штучної освітленості на рівні 300 Лк та коефіцієнта природної освітленості не менше 1,5 %. Рівномірний розподіл світлового потоку та ідентичність яскравості екранів з навколишнім фоном мінімізують напруження периферійного зору та запобігають розвитку передчасної втоми в умовах тривалого виконання складних інженерних завдань.



### 8.4 Розробка заходів з поліпшення умов охорони праці

Проведений аналіз санітарно-гігієнічного стану лабораторії дозволяє констатувати відповідність геометричних параметрів приміщення, показників мікроклімату та акустичного фону встановленим державним стандартам. Основним чиником потенційного зниження працездатності дослідника визначено психофізіологічний фактор, зумовлений високою когнітивною складністю

квантування нейромережевих моделей та налагодження низькорівневого коду для архітектури Xtensa LX7. Мінімізація негативного впливу інтелектуального перенапруження досягається через суворе доотримання регламентованого режиму праці та відпочинку, а також створення сприятливої психологічної атмосфери в інженерному колективі.

Організація робочого місця розробника Edge AI рішень базується на принципах ергономіки, що передбачає раціональне розміщення лабораторних стендів з модулями та контрольно-вимірювальної апаратури. Важливою складовою безпеки є регулярне наочне ознайомлення персоналу зі шляхами евакуації згідно з затвердженим планом, який повинен бути розміщений на доступному для огляду місці. Програма цивільного захисту передбачає включення до складу аптечок першої допомоги засобів для проведення штучного дихання, зокрема масок-клапанів, що є критичним при наданні допомоги у разі виникнення непередбачуваних ситуацій у закритих приміщеннях обчислювальних центрів.

Електротехнічна безпека при роботі з налагоджувальними платами та силовими блоками живлення забезпечується шляхом регулярної перевірки параметрів захисного заземлення та занулення. Вимірювання опору заземлювального ланцюга проводиться з періодичністю, встановленою нормами ПТЕЕС, для запобігання накопиченню статичного заряду на корпусах пристроїв. Розподільні щити лабораторії оснащуються спеціалізованими розетками з інтегрованими заземлюючими контактами, а всі прилади, що функціонують під напругою понад 36 В, підлягають обов’язковому заземленню.

З огляду на специфічний ризик виникнення фібриляції шлуночків серця при випадковому контакті з відкритими струмоведучими частинами обладнання, доцільним є оснащення лабораторії автоматичним зовнішнім дефібрилятором. Ефективність даного заходу гарантується лише за умови попередньої підготовки персоналу до роботи з реанімаційним обладнанням. Запропонований комплекс заходів дозволяє нейтралізувати дію ідентифікованих шкідливих чинників та

забезпечити високу надійність функціонування системи в умовах тривалої інженерної експлуатації.

### 8.5 Розрахункова частина

Розрахунок параметрів штучного освітлення лабораторії TinyML виконано за методом коефіцієнта використання світлового потоку. Обрана методика дозволяє врахувати не лише пряме випромінювання світильників, а й відбиття енергії від внутрішніх поверхонь приміщення шириною 5 м, довжиною 6,2 м та висотою 3,4 м. Для забезпечення комфортних умов при монтажі мікросхем ESP32-S3 та розробці програмного коду встановлено нормовану мінімальну освітленість на рівні 300 Лк.

Розрахункову площу об’єкта визначено як добуток його лінійних розмірів, що становить 31 м². Оскільки в процесі експлуатації спостерігається поступове зниження ефективності випромінювачів через запилення та деградацію світлодіодів, у розрахунок введено коефіцієнт запасу 1,5 та поправковий коефіцієнт нерівномірності 1,1. Параметри відбиття стін та стелі прийняті на рівні 50%, що відповідає типовому світлому оздобленню сучасних обчислювальних центрів.

Для визначення інтегральних характеристик середовища обчислено індекс приміщення, який при висоті підвісу світильників 3 м складає 0,43. Відповідно до отриманого індексу та обраного типу ламп встановлено коефіцієнт використання світлового потоку, що дорівнює 0,23. Загальний розрахунковий світловий потік, необхідний для створення нормованого середовища у приміщенні, де працює 5 осіб, становить 66717 Лм.

За апаратну основу системи освітлення обрано світлодіодні панелі PL PFM 600 потужністю 30 Вт із номінальним світловим потоком 3000 Лм кожна. Шляхом зіставлення сумарної потреби об’єкта в освітленні з одиничною потужністю пристрою визначено необхідну кількість одиниць обладнання. Результат обчислень

вказує на потребу у 22,18 джерелах світла, що при округленні у більшу сторону вимагає встановлення 23 світлодіодних світильників.

**Висновки до розділу**

Комплексний аналіз умов праці інженера-програміста підтвердив критичну роль ергономічних та санітарно-гігієнічних факторів у забезпеченні стабільності розробки ІТ-проєктів. Ідентифікація шкідливих чинників, таких як електромагнітне випромінювання мікроконтролерів ESP32-S3 та зорова напруга, дозволила сформувати дієву стратегію захисту персоналу. Математичне обґрунтування системи освітлення гарантує підтримку зорового комфорту, тоді як розроблені заходи з електробезпеки мінімізують технічні ризики при експлуатації обладнання. Реалізація запропонованих рішень сприяє не лише збереженню здоров'я спеціалістів, а й підвищенню загальної продуктивності праці за рахунок оптимізації робочого середовища.

# 9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, розроблене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначене для функціонування апаратно-програмного комплексу виявлення атак на рівні IoT-пристроїв. В межах України спостерігається дефіцит вітчизняних розробок у галузі периферійних обчислень (Edge AI) для захисту критичної інфраструктури, що зумовлює актуальність даного дослідження.

У випускній кваліфікаційній роботі наведено теоретичне узагальнення та розв'язання науково-технічного завдання щодо впровадження легковажних нейронних мереж у вбудовані системи. Досягнення поставленої мети базувалося на послідовному виконанні таких етапів:

— Проведено аналітичний огляд існуючих систем виявлення вторгнень та архітектурних рішень для інтернету речей.

— Досліджено методи оптимізації нейромережевих моделей TinyML для роботи в умовах обмежених апаратних ресурсів мікроконтролерів.

— На основі отриманих результатів створено програмну реалізацію комплексу детекції аномалій трафіку безпосередньо на периферійному вузлі.

Розроблені алгоритми інференсу, адаптовані під векторні інструкції архітектури Xtensa LX7, дозволяють успішно вирішувати завдання ідентифікації кіберзагроз у реальному часі з мінімальною латентністю. У ході аналізу предметної галузі було ідентифіковано ключові об'єкти взаємодії, визначено їхні функціональні характеристики та побудовано математичну модель детектора атак.

Розроблене програмне забезпечення вирізняється ергономічним та інтуїтивно зрозумілим інтерфейсом адмін-панелі, що забезпечує легкість освоєння продукту персоналом без специфічної підготовки у сфері машинного навчання. При створенні системи застосовано об'єктно-орієнтований підхід та сучасні парадигми проектування інтелектуальних систем.

Програмний комплекс реалізовано з використанням мови високого рівня Python для керуючого інтерфейсу та оптимізованого коду для середовища ESP-IDF. Це дозволило досягти максимальної ефективності обробки тензорних даних, мінімізувати терміни розробки та знизити сукупні витрати на реалізацію проєкту. Система призначена для роботи у зв'язці з кросплатформеним ПЗ під управлінням Windows 10/11 та безпосередньо на апаратних модулях ESP32-S3.

Для забезпечення безкомпромісного рівня безпеки в роботі запропоновано та реалізовано комплекс апаратних засобів захисту, зокрема механізми Secure Boot V2 та шифрування пам'яті за стандартом AES-256 XTS. Це унеможливлює модифікацію прошивки та забезпечує конфіденційність оброблюваних даних.

В цілому створене апаратно-програмне забезпечення підтверджує коректність обраних проектних рішень та повністю відповідає вимогам технічного завдання. Розробка має значний потенціал для подальшого масштабування, включаючи можливість донавчання моделей для виявлення специфічних промислових загроз. Проведене маркетингове та економічне обґрунтування підтвердило високу інвестиційну привабливість проєкту та визначило ключові фактори його ринкового успіху.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Смірнова Т. В., Конопліцька-Слободенюк О. К., Буравченко К. О., Смірнов С. А. та ін. «Дослідження технологій забезпечення кібербезпеки хмарних сервісів IaaS, PaaS та SaaS». *Кібербезпека: освіта, наука, техніка*. 2024. № 4 (24). С. 6–27 .

2. Smirnov O., Sydorenko V., Aleksander M., Zhyharevych O., Yenchev S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings*. 2023. Vol. 3530. P. 256–265.

3. Smirnov A. A., Kuznetsov A. A., Danilenko D. A., Berezovsky A. «The statistical analysis of a network traffic for the intrusion detection and prevention systems». *Telecommunications and Radio Engineering*. 2015. Vol. 74, Issue 1. P. 61–78.

4. Smirnova T., Gnatyuk S., Yudin O., Sydorenko V., Polozhentsev A. «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». *CEUR Workshop Proceedings*. 2022. Vol. 3156. P. 390–399.

5. Смірнов О. А., Смірнова Т. В., Якименко Н. М., Смірнов С. А. та ін. «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах». *Системи управління, навігації та зв'язку*. 2022. № 3 (69). С. 93–98.

6. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». CEUR Workshop Proceedings Volume 3156, 2022, Pages 390-399.

7. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

8. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої

функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

9. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

10. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

11. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». *Communications in Computer and Information Science*, 2021, vol 1486. Springer, Cham. pp 169-184.

12. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

13. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

14. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

15. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In:

Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

16. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

17. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

18. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645.

19. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

20. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

21. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

22. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

23. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling

numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

24. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

25. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

26. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

27. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

28. Smirnov A.A., Kuzne tsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

29. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95.

30. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного*

*технологічного університету. Технічні науки.* №4. С. 103-110. 2020.

31. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка*. № 3(7). С. 43-62. 2020.

32. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

33. О.А. Смірнов, П.С. Усік, «дослідження перспектив використання технологічних рішень в мережах 5g» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

34. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». *Центральноукраїнський науковий вісник. Технічні науки.* № 2(33). с. 161-172, 2019.

35. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

36. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

37. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139.

38. Дрєєва Г.М., Смірнов О.А., Дрєєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.

39. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дрєєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

40. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Конопліцька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

41. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

42. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018.

43. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Алгоритми формування безлічі маршрутів передачі метаданих у антивірусні хмарні системи. Збірник наукових праць "Системи обробки інформації". – Випуск 5 (142). – Х.: ХУПС – 2016. – С. 148-152.

44. Смірнов О.А., Смірнов С.А. Дідик А.К., Дрєєв О.М. Моделі системи нейромережевих експертів безпечної маршрутизації у хмарних антивірусних системах. Збірник наукових праць "Системи обробки інформації". – Випуск 3 (140). – Х.: ХУПС – 2016. – С. 36-39.

45. Warden P., Situnayake D. TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. O'Reilly Media, 2020. 504 p.

46. Смірнов О. А., Дичка І. А. Методи побудови захищених систем моніторингу критичних інфраструктур. Кропивницький: ЦНТУ, 2023. 195 с.

47. Кузнєцов О. О., Євсєєв С. П., Корольов Р. В. Методи та засоби криптографічного захисту інформації: навч. посібник. Харків: ХНУРЕ, 2020. 280 с.

48. Гнатюк С. О., Смірнова Т. В. Захист інформації в хмарних

обчислювальних середовищах: монографія. Київ: НАУ, 2021. 240 с.

49. Espressif Systems. ESP32-S3 Series Datasheet. v1.6. 2023. 85 p.

50. Bakhshi T. «State-of-the-art AI-based intrusion detection systems in IoT: A review». IEEE Access. 2021. Vol. 9. P. 12826–12845.

51. Sahu N., Sharma S. K. «Edge Computing and AI for 5G-IoT Infrastructure: A Survey». Journal of Network and Computer Applications. 2022. Vol. 203.

52. Смірнова Т. В., Гнатюк С. О., Юдін О. К. Методологія оцінки рівня захищеності інформаційно-комунікаційних систем. Київ: Вид-во «Наукова думка», 2022. 210 с.

53. Abadi M., et al. «TensorFlow Lite: Machine Learning on Mobile and IoT Devices». IEEE Software. 2021. Vol. 38, Issue 3. P. 101-109.