Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

Галюс Вікторії Юріївни

на здобуття ступеня вищої освіти Бакалавра

Програмно-апаратна система криптографічно-захищеної комунікації в корпоративній мережі

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

Шифр КРБКБ.220105.22.01.04 ПЗ

Виконав студентка 4 курсу група КБ-22-1 Вікторія ГАЛЮС
Керівник д-р техн. наук, професор Михайло КАСЯНЧУК
Нормоконтролер д-р філософії Наталія ПЕТЛЯК

До захисту допускаю:
Завідувач кафедри кібербезпеки Юрій КЛЬОЦ
8 06 2026 р.

Хмельницький 2026

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Кібербезпеки

Рівень вищої освіти Бакалавр

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека

Освітня програма Кібербезпека

ЗАТВЕРДЖУЮ

Завідувач кафедри кібербезпеки

Юрій КЛЬОЦ

09 лютого 2026 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Галюс Вікторії Юріївні

1 Тема роботи Програмно-апаратна система криптографічно-захищеної комунікації в корпоративній мережі

Керівник роботи д-р техн. наук, професор Касянчук М.М.

Затверджено наказом ректора університету від 8 січня 2026 № 7

2 Строк подання студентом кваліфікаційної роботи на кафедру 25 травня 2026

3 Вихідні дані до роботи Дослідити існуючі методи криптографічного захисту даних під час передачі в комп'ютерних мережах. Проаналізувати актуальні загрози інформаційній безпеці в корпоративних інтранет-мережах. Розробити модель загроз та модель порушника системи. Спроєктувати архітектуру програмно-апаратного криптографічного шлюзу на базі Raspberry Pi 4B. Реалізувати захищений канал зв'язку з використанням протоколу WireGuard. Розробити схему ізоляції ключового матеріалу та механізми його планової ротації і знищення. Реалізувати програмно-апаратний прототип системи з підключенням через інтерфейс USB Ethernet Gadget. Провести тестування прототипу та оцінити його продуктивність і захищеність.

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити) Вступ. Аналіз методів криптографічного захисту в мережах, нормативної бази ТЗІ та існуючих технічних рішень для захисту каналів зв'язку. Обгрунтування вибору апаратної платформи і базового ПЗ. Постановка задачі проєктування, побудова моделей загроз та порушника. Розробка архітектури системи, механізмів управління її компонентами, а також схем ізоляції, планової ротації, відкликання та знищення ключових даних. Формування технічних вимог, вибір метрик, розробка та тестування програмно-апаратного прототипу системи. Розробка настанов щодо експлуатації та висновки.

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень) Структурна схема та модель загроз системи. Функціональна схема та архітектура ізоляції ключів. Схема алгоритму управління ключами та результати тестування.

6 Консультанти розділів кваліфікаційної роботи

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

7 Дата видачі завдання 09 лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
Вибір і затвердження теми кваліфікаційної роботи	Січень-Лютий	
Ознайомлення з предметною областю	Лютий	
Дослідження існуючих рішень	Лютий	
Постановка задачі	Березень	
Визначення загальних принципів рішення задачі	Березень	
Деталізація принципів рішення задачі	Квітень	
Розробка проєктних рішень	Квітень	
Апробація проєктних рішень	Травень	
Оформлення пояснювальної записки згідно вимог	Травень	
Оформлення графічної частини	Травень	
Захист КР	Червень	

Студентка _____ Вікторія ГАЛЮС

Керівник кваліфікаційної роботи _____ Михайло КАСЯНЧУК

АНОТАЦІЯ

Тема кваліфікаційної роботи: Програмно-апаратна система криптографічно-захищеної комунікації в корпоративній мереж

Автор роботи: Галюс Вікторія Юріївна

Керівник роботи: д-р техн. наук, проф. Касянчук Михайло Миколайович

Пояснювальна записка: 85 с., 20 рис., 19 табл., 53 джерела

Графічна частина: 3 плакати

Ключові слова: інформаційна безпека, криптографічний шлюз, апаратна ізоляція, WireGuard, Raspberry Pi 4B, USB Gadget, захист каналів зв'язку, модель загроз, ротація ключів, ефемерні сховища.

У кваліфікаційній роботі досліджено вразливості програмних VPN-клієнтів на рівні хост-ОС та обґрунтовано доцільність апаратної ізоляції криптографічних функцій і ключового матеріалу від потенційно скомпрометованої робочої станції.

Спроєктовано архітектуру захищеного вузла на базі Raspberry Pi 4B із підключенням до ПК через віртуальний інтерфейс USB Gadget (режими RNDIS/CDC-ECM). Розроблено багаторівневе сховище криптосекретів, що поєднує ефемерну пам'ять RAM (tmpfs) та енергонезалежні розділи із шифруванням LUKS.

Реалізовано Python-модуль для управління життєвим циклом ключів (генерація, планова ротація, екстрене відкликання) за протоколом WireGuard (ChaCha20-Poly1305). Емпірична верифікація пропускної здатності та затримок підтвердила високу ефективність і зручність експлуатації розробленого прототипу.

25.05.2026 Галюс В.Ю.

ABSTRACT

Subject of qualification work: Hardware-Software System for Cryptographically Secured Communication in a Corporate Network

Author: Halius Viktoriia Yuriivna

Head of work: Doctor of Technical Sciences, Professor Kasianchuk Mykhailo Mykolaiovych

Explanatory note: 85 pages, 20 figures, 19 tables, 53 sources

Graphic part: 3 posters

Keywords: information security, cryptographic gateway, hardware isolation, WireGuard, Raspberry Pi 4B, USB Gadget, communication channel protection, threat model, key rotation, ephemeral storage.

The qualification work investigated the vulnerabilities of software VPN clients at the host OS level and justified the feasibility of hardware isolation of cryptographic functions and key material from a potentially compromised workstation.

A secure node architecture based on Raspberry Pi 4B has been designed with connection to a PC via a virtual USB Gadget interface (RNDIS/CDC-ECM modes). A multi-level cryptographic secret storage has been developed, combining ephemeral RAM (tmpfs) and non-volatile partitions with LUKS encryption.

A Python module for key lifecycle management (generation, scheduled rotation, emergency revocation) using the WireGuard protocol (ChaCha20-Poly1305) has been implemented. Empirical verification of throughput and latency confirmed the high efficiency and ease of use of the developed prototype.

25.05.2026 Галиус В.Ю.

ЗМІСТ

ВСТУП

Стрімка еволюція цифрових технологій та повсюдна інтеграція хмарних обчислень у корпоративні структури призвели до докорінної зміни ландшафту кіберзагроз. Традиційна модель захисту інформації, що базувалася на концепції «захищеного периметра», де внутрішня мережа організації вважалася апріорі довіреною, на сьогодні визнана недієздатною. Сучасні цілеспрямовані атаки (далі – APT), методи соціальної інженерії та стрімке зростання внутрішніх порушників демонструють, що зловмисник може перебувати всередині мережі протягом тривалого часу, залишаючись непоміченим для класичних засобів моніторингу. У відповідь на ці виклики світова спільнота з кібербезпеки перейшла до парадигми архітектури нульової довіри (далі - ZTA), ключовим принципом якої є обов’язкова перевірка та шифрування кожної взаємодії між вузлами, незалежно від їхнього фізичного чи логічного розташування в інфраструктурі.

Проте реалізація надійного шифрування на рівні кінцевих пристроїв (робочих станцій користувачів) стикається з фундаментальною проблемою «спільного середовища». Коли криптографічні операції виконуються безпосередньо в операційній системі загального призначення (Windows, macOS або Linux), вони стають вразливими до компрометації самого ядра ОС. Наявність шкідливого програмного забезпечення, такого як кейлогери або засоби зчитування оперативної пам’яті (memory scrapers), дозволяє зловмиснику перехоплювати відкриті дані ще до моменту їхнього зашифрування або, що є критичнішим, викрадати приватні ключі доступу. Таким чином, навіть використання найсучасніших алгоритмів стійкого шифрування не гарантує безпеки, якщо середовище, в якому вони функціонують, є потенційно «гнилим» або скомпрометованим.

Виходом із цієї ситуації є застосування програмно-апаратних засобів, що забезпечують фізичну ізоляцію криптографічного ядра від основного комп’ютера користувача. Традиційні апаратні модулі безпеки (далі - HSM) є ідеальним рішенням з точки зору захищеності, проте їхня висока вартість та складність налаштування роблять їх недоступними для масового впровадження на кожному

робочому місці в межах корпоративної мережі. Дана робота пропонує альтернативний шлях: створення спеціалізованого криптографічного шлюзу на базі доступних мікрокомп’ютерних платформ. Такий підхід дозволяє винести процеси генерації ключів, автентифікації та встановлення шифрованого тунелю на окремий апаратний пристрій, який виконує роль довіреного посередника.

Варто наголосити, що запропонована система не претендує на роль абсолютного захисту від атак державного рівня чи складного лабораторного аналізу, проте вона створює суттєвий бар’єр для типових корпоративних загроз. Використання протоколу WireGuard як транспортного рівня дозволяє досягти високої швидкості обробки трафіку при мінімальному об’ємі програмного коду, що значно полегшує аудит безпеки. Особлива увага в роботі приділяється розділенню рівнів передачі даних (Data Plane) та управління (Control Plane), що є критично важливим для побудови масштабованих корпоративних систем з централізованим управлінням політиками довіри та життєвим циклом ключів.

Актуальність теми дослідження зумовлена гострою потребою вітчизняних підприємств та державних установ у недорогих, але ефективних засобах захисту внутрішніх комунікацій, які б відповідали вимогам технічного захисту інформації та могли бути оперативно розгорнуті в умовах гібридних загроз. Об’єктом дослідження є процеси забезпечення конфіденційності та цілісності інформаційних потоків у корпоративних мережах. Предметом дослідження виступають архітектурні принципи, алгоритми та програмно-апаратні засоби побудови ізольованих систем криптографічного захисту.

Метою роботи є проєктування та реалізація програмно-апаратного прототипу системи криптографічно захищеної комунікації, яка забезпечує ізоляцію ключових даних від кінцевої робочої станції та прозоре шифрування трафіку в межах корпоративного інтранету.

Для досягнення цієї мети в роботі поставлено та вирішено такі завдання:

– виконати аналіз існуючих методів захисту каналів зв'язку та обґрунтувати переваги програмно-апаратної ізоляції криптофункцій;

– провести порівняльний огляд сучасних протоколів тунелювання (IPsec, OpenVPN, WireGuard) та апаратних платформ для побудови захищених шлюзів;

– розробити модель загроз та модель порушника, орієнтовану на умови корпоративної мережі;

– спроєктувати архітектуру системи, включаючи механізми управління ключами (PKI) та схеми їхньої планової ротації;

– реалізувати програмно-апаратний прототип на базі Raspberry Pi 4 з використанням інтерфейсу USB Ethernet Gadget для підключення до ПК;

– провести експериментальну оцінку продуктивності системи та перевірити її стійкість до типових мережевих атак.

Практична цінність результатів роботи підтверджується створенням діючого прототипу криптографічного шлюзу на базі Raspberry Pi 4B, який забезпечує шифрування мережевого трафіку та ізоляцію ключового матеріалу від операційної системи хост-комп’ютера. Розроблений пристрій може бути використаний для розгортання захищених каналів зв’язку у підрозділах з підвищеними вимогами до безпеки інформації при мінімальних витратах на впровадження. Крім того, відкрита архітектура розробленого програмного забезпечення дозволяє гнучко адаптувати комплекс до специфічних потреб конкретного підприємства без потреби у додаткових ліцензійних витратах. Впровадження таких автономних засобів також значно спрощує процес адміністрування мережі завдяки реалізації принципу швидкого розгортання.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ, АКТУАЛЬНИХ ДОСЯГНЕНЬ ТА КРАЩИХ ПРАКТИК ЗАХИСТУ ЗА ТЕМОЮ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1.1 Огляд методів криптографічного захисту даних під час передачі в комп'ютерних мережах

Забезпечення безпеки інформаційної взаємодії в умовах глобальної цифровізації корпоративних процесів є однією з найбільш складних та пріоритетних задач сучасної кібербезпеки. Стрімкий розвиток мережевих технологій призвів до того, що дані, які раніше зберігалися в ізольованих контурах, тепер постійно перебувають у стані транзиту через відкриті або потенційно ворожі мережеві сегменти. У таких умовах криптографічний захист інформації розглядається не просто як додатковий рівень безпеки, а як фундаментальний фундамент, що забезпечує виконання тріади кібербезпеки: конфіденційності, цілісності та доступності даних [1].

Криптографічні методи захисту інформації в комп'ютерних мережах базуються на застосуванні математичних перетворень, які дозволяють приховати зміст повідомлення від несанкціонованого ознайомлення та гарантувати, що дані не були змінені під час передачі. Класифікація сучасних криптосистем традиційно проводиться за типом використання ключів, поділяючи їх на симетричні та асиметричні системи, кожна з яких виконує специфічну роль у побудові захищеного каналу зв'язку [1].

Симетричне шифрування становить основу продуктивності сучасних VPN-тунелів та захищених протоколів. Основним критерієм вибору симетричного алгоритму для корпоративної мережі є баланс між швидкістю обробки та стійкістю до криптоаналізу. Найбільш розповсюдженим стандартом у світі є AES (Advanced Encryption Standard), який працює з блоками даних фіксованого розміру (128 біт) та ключами довжиною до 256 біт [10]. Проте, незважаючи на високу стійкість, AES у програмній реалізації може створювати значне навантаження на центральний процесор, якщо в архітектурі відсутня апаратна підтримка інструкцій AES-NI. Це спонукає розробників сучасних систем до впровадження альтернативних потокових шифрів. Для наочного розуміння відмінностей у

підходах до управління ключами, принципова схема функціонування симетричних та асиметричних методів наведена на рисунку 1.1.

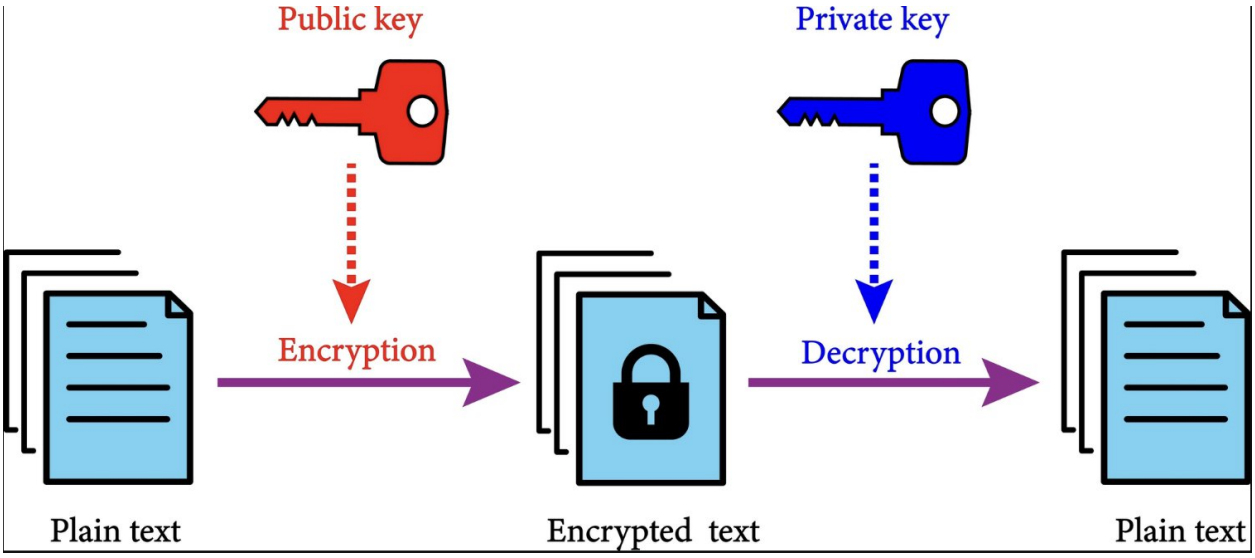


Рисунок 1.1 – Схематичне порівняння симетричних та асиметричних криптосистем

Одним із найбільш перспективних рішень у цій галузі є алгоритм ChaCha20, який у поєднанні з кодом автентифікації Poly1305 утворює конструкцію AEAD (Authenticated Encryption with Associated Data). На відміну від блочних шифрів, ChaCha20 демонструє виняткову продуктивність на пристроях з архітектурою ARM (наприклад, Raspberry Pi), забезпечуючи високу швидкість шифрування без необхідності використання спеціалізованих апаратних прискорювачів [11]. Режим AEAD дозволяє одночасно шифрувати корисне навантаження пакетів та обчислювати криптографічну мітку, яка гарантує, що жоден біт інформації не був змінений зловмисником під час передачі, що радикально підвищує стійкість системи до атак типу «Man-in-the-Middle» (MITM).

Асиметрична криптографія, або криптографія з відкритим ключем, відіграє критичну роль на етапі ініціалізації з'єднання та автентифікації вузлів. Традиційні методи, засновані на алгоритмі RSA, вимагають використання ключів великої довжини (від 3072 біт), що негативно впливає на швидкість встановлення сесії та об’єм службового трафіку [12]. Сучасним стандартом «де-факто» для вбудованих та високонавантажених систем стала криптографія на еліптичних кривих (Elliptic

Curve Cryptography, ECC). Математична стійкість ECC базується на складності розв’язання задачі дискретного логарифмування в групі точок еліптичної кривої, що дозволяє використовувати значно менші ключі при збереженні ідентичного рівня безпеки.

Зокрема, використання кривої Curve25519 для протоколів обміну ключами (Diffie-Hellman на еліптичних кривих) забезпечує високу швидкість обчислень та стійкість до сучасних методів криптоаналізу [13]. Важливою властивістю таких протоколів є забезпечення Perfect Forward Secrecy (PFS) механізму, за якого компрометація довгострокових ключів пристрою не дозволяє зловмиснику розшифрувати сесії, що були записані раніше. Це досягається шляхом генерації унікальних ефемерних ключів для кожної нової комунікаційної сесії.

Окрім безпосередньо шифрування, критично важливою складовою мережевого захисту є контроль цілісності та автентичності через використання хеш-функцій та цифрових підписів. Сучасні мережеві протоколи, такі як WireGuard, відмовляються від застарілих та громіздких архітектур на користь мінімалістичного набору криптографічних примітивів, що дозволяє не лише підвищити швидкість роботи, а й значно спростити проведення незалежного аудиту сирцевого коду на наявність вразливостей або недекларованих можливостей [6].

Підсумовуючи огляд методів, слід зазначити, що побудова надійної програмно-апаратної системи захищеної комунікації в корпоративному середовищі вимагає комплексного підходу. Необхідно поєднувати стійкі алгоритми симетричного шифрування для захисту основного потоку даних з ефективними методами на базі еліптичних кривих для управління ключами та автентифікації. Саме такий стек технологій дозволяє реалізувати модель ізольованої криптографічної обробки, де найбільш критичні операції виносяться на окремий апаратний пристрій, мінімізуючи ризики, пов'язані з компрометацією основної операційної системи користувача [9].

1.2 Аналіз нормативно-правового забезпечення та стандартів у сфері технічного захисту інформації

Нормативно-правове регулювання у сфері технічного захисту інформації (далі - ТЗІ) та криптографічного захисту інформації (далі - КЗІ) є багаторівневою структурою, яка визначає не лише технічні параметри систем, а й юридичну легітимність їхнього використання в межах держави. Для розробки програмно-апаратної системи, що претендує на використання в корпоративному секторі, критично важливо розуміти ієрархію нормативних актів, оскільки будь-яка помилка на етапі проектування з точки зору відповідності нормативним вимогам може зробити систему непридатною для впровадження на об'єктах критичної інфраструктури або в державних установах.

Верхній рівень правового регулювання в Україні представлений базовими законами. Закон України «Про захист інформації в інформаційно-комунікаційних системах» є фундаментом, який встановлює обов'язковість захисту інформації, що є власністю держави, або інформації, захист якої передбачено законом (наприклад, персональні дані) [5]. Згідно зі статтею 8 цього Закону, захист інформації має забезпечуватися шляхом створення Комплексної системи захисту інформації (КСЗІ). Для розробника програмно-апаратного рішення це означає, що пристрій не може існувати сам по собі, адже він має стати частиною загальної архітектури безпеки, яка проходить атестацію.

Закон України «Про основні засади забезпечення кібербезпеки України» вводить поняття об'єктів критичної інфраструктури та встановлює підвищені вимоги до їхнього захисту [14]. Це безпосередньо стосується теми диплома, оскільки корпоративні мережі енергетичних, фінансових або транспортних компаній підпадають під дію цього закону. Він вимагає використання засобів захисту, які мають підтверджену відповідність (сертифікацію), що змушує розробників орієнтуватися на національні критерії оцінки безпеки інформаційних технологій (НД ТЗІ 2.5-004-99) [15].

Важливим аспектом є нормативна база Державної служби спеціального зв'язку та захисту інформації (Держспецзв'язку). Саме ця служба видає

нормативні документи (НД ТЗІ), які деталізують технічні вимоги. Наприклад:

– НД ТЗІ 3.7-003-05 визначає порядок проведення робіт зі створення КСЗІ;

– НД ТЗІ 2.5-008-02 встановлює вимоги до засобів захисту від несанкціонованого доступу;

– Положення про державну експертизу у сфері ТЗІ описує процедуру отримання експертного висновку на засіб захисту.

Для програмно-апаратних систем (Hardware Security Modules або Crypto Appliances) критичне значення мають міжнародні стандарти, оскільки вони визначають фізичну стійкість пристрою. Стандарт FIPS 140-3 [4], розроблений американським NIST, де-факто є світовим стандартом для криптографічних модулів. Він розділяє вимоги на чотири рівні безпеки:

– рівень 1, мінімальні вимоги, що стосуються лише програмної реалізації алгоритмів;

– рівень 2, вимагає наявності фізичних доказів втручання (tamper-evidence), наприклад, спеціальних пломб або покриттів, які руйнуються при спробі відкрити корпус;

– рівень 3, вимагає механізмів активного реагування на розтин (tamper-resistance), наприклад, миттєве стирання критичних параметрів (криптографічних ключів) при виявленні фізичного зламу;

– рівень 4, забезпечує захист від складних атак з використанням параметрів навколишнього середовища (зміна напруги, температури), що зазвичай притаманно системам військового призначення.

У контексті міжнародної інтеграції неможливо ігнорувати стандарт ISO/IEC 15408 «Common Criteria» [16]. Цей стандарт використовує рівні довіри до оцінки (EAL - Evaluation Assurance Level) від EAL1 до EAL7. Більшість комерційних засобів захисту мереж (Firewalls, VPN-gateways) сертифікуються за рівнями EAL4 або EAL4+, що означає ретельне тестування та аналіз сирцевого коду розробником і незалежною лабораторією. Проектування системи на базі Raspberry Pi з відкритим кодом WireGuard дозволяє теоретично досягти високих рівнів довіри через прозорість алгоритмів та можливість повного аудиту коду.

Окрему увагу слід приділити нормам захисту персональних даних. В

Україні діє Закон «Про захист персональних даних» [17], який гармонізований із європейським регламентом GDPR [2]. Стаття 32 GDPR («Security of processing») прямо вимагає впровадження шифрування як одного з основних технічних заходів захисту. Це означає, що використання розроблюваної системи криптографічного захисту в корпоративному секторі є не просто «бажаним бонусом», а юридичною необхідністю для виконання вимог щодо захисту даних клієнтів та співробітників.

Для наочності аналізу нормативного забезпечення доцільно порівняти національні та міжнародні підходи, які наведені в таблиці 1.1 нижче.

Таблиця 1.1 - Порівняльна таблиця стандартів КЗІ та ТЗІ

Сфера регулювання	Національний стандарт (Україна)	Міжнародний стандарт / Регламент
Управління безпекою	ДСТУ ISO/IEC 27001:2015	ISO/IEC 27001:2022
Криптоалгоритми	ДСТУ 7624:2014 (Калина)	AES (FIPS 197), ChaCha20 (RFC 8439)
Електронний підпис	ДСТУ 4145-2002	ECDSA, EdDSA (RFC 8032)
Оцінка засобів захисту	НД ТЗІ 2.5-004-99	Common Criteria (ISO/IEC 15408)
Апаратна безпека	Вимоги ДССЗЗІ до засобів КЗІ	FIPS 140-3
Захист даних	ЗУ "Про захист персональних даних"	GDPR (EU 2016/679)

Завершуючи детальний аналіз, варто підкреслити роль технічних регламентів та галузевих стандартів. Наприклад, у банківському секторі діють постанови Національного банку України (наприклад, Постанова №95), які встановлюють специфічні вимоги до шифрування каналів зв'язку між філіями банків [5]. У промислових мережах (OT/ICS) діє стандарт IEC 62443, який

акцентує увагу на цілісності команд управління [2].

Таким чином, розробка програмно-апаратної системи криптографічного захисту в рамках даної роботи повинна враховувати цей багатогранний нормативний ландшафт. Гнучкість вибраної платформи та протоколів дозволяє адаптувати систему під вимоги як вітчизняного законодавства (через потенційну реалізацію ДСТУ), так і міжнародних стандартів безпеки, що робить проект актуальним для широкого кола корпоративних користувачів, які прагнуть забезпечити комплаєнс у сфері кібербезпеки.

1.3 Порівняльний аналіз існуючих технічних рішень та програмних засобів побудови захищених каналів зв'язку

Аналіз сучасного ринку засобів захисту інформації свідчить про глибоку диференціацію методів побудови захищених каналів, проте кожне з існуючих рішень змушує проектанта йти на компроміс між безпекою, керованістю та вартістю [1]. Для об’єктивного порівняння необхідно розглядати не лише програмні алгоритми, а й середовища, у яких вони функціонують, оскільки стійкість криптосистеми визначається стійкістю її найслабшої ланки. У більшості корпоративних сценаріїв такою ланкою є кінцева робоча станція користувача (Endpoint), де програмний захист стикається з фундаментальними обмеженнями архітектури загального призначення [3].

Програмні засоби реалізації VPN-тунелів на сьогодні є найпоширенішим класом рішень завдяки їхній нульовій вартості апаратного впровадження. Протокол OpenVPN, який довгий час вважався галузевим стандартом, базується на архітектурі TLS/SSL і пропонує надзвичайну гнучкість у налаштуванні. Його головна перевага - це здатність працювати через будь-які порти та протоколи (TCP/UDP), що дозволяє легко обходити міжмережеві екрани. Проте зворотним боком цієї гнучкості є монолітна та надмірно складна кодова база [18]. OpenVPN виконується у просторі користувача (user-space), що створює значне навантаження на центральний процесор через постійні перемикання контексту та

копіювання пакетів між ядром і додатком. З точки зору безпеки, величезний об’єм сирцевого коду (сотні тисяч рядків) робить практично неможливим його повний аудит, що створює ризики наявності не виявлених вразливостей, подібних до критичних помилок у бібліотеці OpenSSL.

Альтернативою є IPsec, який інтегрований безпосередньо в мережевий стек більшості сучасних операційних систем. Він працює на рівні ядра, що забезпечує значно вищу пропускну здатність у порівнянні з OpenVPN. Однак IPsec є надзвичайно складним у конфігурації через велику кількість фаз узгодження параметрів (IKE/IKEv2), що часто призводить до «людського фактора» помилок у налаштуваннях, які знижують реальну стійкість каналу. Більше того, IPsec часто сприймається як «важкий» протокол, який погано працює в умовах нестабільного зв’язку та вимагає статичних IP-адрес для надійної роботи.

Поява протоколу WireGuard стала точкою перелому в еволюції програмних засобів захисту. Його ключова філософія - це мінімалізм та використання лише найсучасніших криптографічних примітивів за замовчуванням (Curve25519, ChaCha20, Poly1305). WireGuard має об’єм коду близько 4000 рядків, що дозволяє невеликій команді фахівців провести його повний формальний аудит за лічені дні [6]. Він працює в просторі ядра, показуючи продуктивність, що наближається до швидкості «сирого» мережевого інтерфейсу. Проте, як і всі програмні VPN, він має критичний недолік у контексті корпоративної безпеки: ключі шифрування зберігаються в оперативній пам'яті комп'ютера. У разі успішної атаки на ядро ОС або через використання вразливостей апаратного забезпечення (наприклад, атаки типу Rowhammer або DMA-атаки), зловмисник може отримати доступ до ключів, незважаючи на всю досконалість протоколу [20].

Апаратні корпоративні рішення від таких гігантів, як Cisco, Fortinet або Check Point, пропонують вищий рівень захисту за рахунок використання спеціалізованих інтегральних схем (ASIC). Ці пристрої міжмережеві екрани нового покоління (далі - NGFW) здатні виконувати шифрування та глибокий аналіз пакетів (DPI) на апаратному рівні, забезпечуючи гігабітні швидкості без затримок. Основна проблема цих систем полягає в концепції «чорної скриньки». Пропрієтарне програмне забезпечення не підлягає зовнішньому аудиту, що

створює ризики наявності «бекдорів», залишених розробником або спецслужбами країни-виробника. Крім того, вартість володіння такими системами включає не лише дороге залізо, а й щорічні ліцензії, що робить їх недоступними для середнього бізнесу або специфічних державних підрозділів [21].

Окрему категорію складають Апаратні модулі безпеки (HSM). Це найбільш стійкі пристрої, які гарантують неможливість експорту приватного ключа навіть при повному фізичному доступі до пристрою. Вони мають сертифікацію FIPS 140-2/3 і призначені для банківського сектора та центрів сертифікації ключів. Проте HSM не є мережевими пристроями у повному розумінні; вони виконують роль «криптографічного співпроцесора». Інтеграція HSM для захисту потокового трафіку в реальному часі вимагає створення складних програмних прошарків, що різко збільшує складність і вартість системи, роблячи її непридатною для захисту звичайної корпоративної комунікації між співробітниками.

Порівняння існуючих технічних рішень дозволяє виокремити декілька критичних параметрів, за якими проводиться вибір у межах даної кваліфікаційної роботи. По-перше, це рівень ізоляції ключового матеріалу у програмних рішеннях він нульовий, в HSM максимальний. По-друге, це прозорість для кінцевого користувача. Більшість апаратних VPN-шлюзів потребують встановлення клієнтського ПЗ на ПК, що знову повертає нас до ризиків компрометації ОС. Детальні результати аналізу та зіставлення характеристик розглянутих засобів наведено у таблиці 1.2.

Підсумовуючи результати порівняльного аналізу, можна констатувати, що на ринку існує незаповнена ніша для рішень, які б забезпечували рівень безпеки, близький до апаратних шлюзів, але за вартістю та простотою використання наближалися до програмних VPN. Більшість існуючих систем або занадто складні для масового впровадження, або ігнорують загрозу компрометації самого комп’ютера, до якого підключений канал зв’язку. Застосування мікрокомп’ютера Raspberry Pi як ізольованого криптографічного шлюзу дозволяє вирішити цю проблему шляхом фізичного відокремлення обробки ключів від середовища користувача [9]. Такий підхід забезпечує надійну «гігієну» ключового матеріалу та прозорість мережевого тунелювання, що робить його ідеальним кандидатом

для реалізації в умовах сучасних корпоративних мереж з високими вимогами до безпеки та обмеженими ресурсами на адміністрування.

Таблиця 1.2 - Порівняльна характеристика засобів криптографічного захисту каналів зв'язку

Параметр порівняння	Програмний VPN (OpenVPN/IPsec)	Корпоративний NGFW (Cisco/Fortinet)	Апаратний HSM (Thales/SafeNet)	Пропонований Crypto-Box (RPi + WG)
Архітектура	Pure Software	Proprietary Hardware	Dedicated Crypto HW	Hybrid (COTS Hardware)
Кодова база	Велика (складна для аудиту)	Закрита (аудит неможливий)	Середня (пропрієтарна)	Мінімальна (Open Source)
Ізоляція ключів	Відсутня (RAM хост-ПК)	Програмна в межах заліза	Повна апаратна	Апаратна (окремий пристрій)
Швидкість розгортання	Миттєва	Тривала (налаштування HW)	Складна інтеграція	Висока (Plug & Play)
Ризик "бекдорів"	Низький (для Open Source)	Високий (Vendor lock-in)	Низький	Мінімальний
Вартість	Низька / Безкоштовно	Дуже висока	Екстремальна	Низька (бюджетна)

Як свідчать наведені у таблиці 1.2 дані, концепція розроблюваного криптографічного шлюзу демонструє найкращий баланс між надійністю апаратної ізоляції ключів та доступністю відкритого програмного забезпечення. Запропонована архітектура ефективно нівелює критичні недоліки як суто програмних VPN-клієнтів, так і дорогих пропрієтарних рішень, що повністю обґрунтовує доцільність її використання як базової платформи для подальшого технічного проєктування системи.

1.4 Обґрунтування вибору апаратної платформи та базового програмного забезпечення для побудови системи

Вибір технологічного стека для створення програмно-апаратного комплексу захисту інформації є фундаментом, що визначає не лише поточну працездатність прототипу, а й його стійкість до специфічних векторів атак, що виникають на межі фізичного та цифрового середовищ. Головною метою при виборі апаратної бази було знаходження балансу між обчислювальною потужністю, енергоефективністю, вартістю та наявністю специфічних апаратних інтерфейсів, необхідних для реалізації концепції «crypto appliance». Після ретельного аналізу існуючих рішень, від мікроконтролерних систем до потужних x86-платформ, було обрано одноплатний комп'ютер Raspberry Pi 4 Model B. Цей вибір обумовлений не лише популярністю платформи, а й глибокими архітектурними перевагами, які критично важливі для криптографічного шлюзу [9].

Серцем системи є SoC Broadcom BCM2711, що містить чотири ядра ARM Cortex-A72 з тактовою частотою 1,5 ГГц. Важливою особливістю цієї архітектури є підтримка набору інструкцій ARMv8 [23], що дозволяє ефективно виконувати сучасні криптографічні операції. На відміну від мікроконтролерів, які часто «захлинаються» на операціях з плаваючою комою або складних математичних перетвореннях на еліптичних кривих, RPi 4B демонструє продуктивність, достатню для шифрування трафіку на швидкостях понад 300 Мбіт/с у реальному часі. Оскільки в криптографії якість ентропії є фундаментальним параметром, наявність апаратного джерела випадковості гарантує, що генеровані сесійні ключі будуть стійкими до атак передбачення, що часто є вразливим місцем у чисто програмних системах [4].

Вирішальною технічною перевагою Raspberry Pi 4B є підтримка режиму USB-C OTG через контролер DWC2. Це дозволяє пристрою функціонувати не лише як хост-контролер, а й як периферійний вузол. Для реалізації нашої системи це означає можливість використання стека USB Gadget [7], зокрема протоколів RNDIS(для Windows) та CDC-ECM (для Linux/macOS). Завдяки цьому «крипто-бокс» розпізнається комп'ютером користувача як

зовнішній мережевий адаптер. Такий підхід забезпечує повну фізичну та логічну ізоляцію ключового матеріалу: операційна система комп'ютера бачить лише новий мережевий інтерфейс і не має прямого доступу до файлової системи або оперативної пам'яті Raspberry Pi, де зберігаються приватні ключі.

Детальні технічні характеристики розглянутих платформ та їх порівняльний аналіз наведено в таблиці 1.3.

Таблиця 1.3 – Технічні характеристики та порівняльний аналіз апаратних платформ

Характеристика	ESP32-S3	Orange Pi 3 LTS	Raspberry Pi 4B	Intel NUC (Mini-PC)
Архітектура	Xtensa Dual-Core	ARM Cortex-A53	ARM Cortex-A72	x86_64
Об'єм RAM	512 KB	2 GB	2 - 8 GB	8 - 32 GB
Ethernet	Зовнішній модуль	1 Gbps	1 Gbps (Native)	2.5 Gbps
USB Gadget (OTG)	Так (обмежено)	Так	Повна підтримка	Рідко / Складно
Апаратний TRNG	Так	Так	Так (стабільний)	Так (RDRAND)
Енергоспоживання	< 1 Вт	3 - 5 Вт	3 - 6 Вт	15 - 45 Вт
Сфера застосування	IoT сенсори	Бюджетні сервери	Крипто-шлюзи	Серверні станції

Вибір програмного забезпечення для системи ґрунтується на принципах мінімалізму та максимальної довіри до коду. Як операційну систему обрано Raspberry Pi OS Lite (Debian Bullseye/Bookworm). Використання "Lite" версії дозволяє виключити з системи графічну підсистему (X11/Wayland), що не лише вивільняє ресурси процесора, а й радикально зменшує поверхню атаки, видаляючи сотні потенційно вразливих пакунків, які не потрібні для роботи

шлюзу.

Основним протоколом тунелювання обрано WireGuard [6]. Його інтеграція безпосередньо в ядро Linux (починаючи з версії 5.6) усуває затримки, пов'язані з перемиканням контексту між простором користувача та простором ядра, що характерно для застарілих рішень типу OpenVPN. WireGuard базується на сучасних криптографічних примітивах, таких як Curve25519 для обміну ключами та ChaCha20-Poly1305 для автентифікованого шифрування [11]. Математична стійкість такого підходу доведена численними аудитами [24], а об’єм коду (близько 4000 рядків) робить його ідеальним для систем, де критично важливою є можливість незалежної перевірки коду на наявність помилок або «бекдорів».

Для наочності порівняємо обраний програмний стек з існуючими альтернативами в розрізі мережевих протоколів, що наведено у таблиці 1.4.

Таблиця 1.4 – Порівняльна характеристика мережевих протоколів захисту інформації

Параметр	OpenVPN	IPsec (IKEv2)	WireGuard
Місце в ОС	User-space	Kernel-space	Kernel-space
Об'єм коду (рядків)	~600,000	~400,000	~4,000
Крипто-стек	Будь-який (OpenSSL)	Складний (Suite B)	Фіксований (Сучасний)
Швидкість (Handshake)	Повільно (TLS)	Середня	Миттєво (1-RTT)
Аудит безпеки	Вкрай важкий	Важкий	Простий
Стійкість до DoS	Низька	Середня	Висока (Cookie mechanism)

Важливим компонентом є також вибір мови програмування для рівня управління (Control Plane). Хоча сам трафік шифрується ядром Linux, логіка керування пристроєм (генерація ключів, автентифікація, оновлення конфігурацій) реалізована на Python 3. Використання бібліотеки cryptography для Python

дозволяє безпечно працювати з ключами на високому рівні абстракції, уникаючи помилок роботи з пам’яттю, які часто трапляються в С-реалізаціях. Це гарантує, що процедура ротації ключів, яка є критичною для безпеки корпоративної мережі, буде виконана надійно та прозоро.

Математичне обґрунтування вибору алгоритмів також грає важливу роль. Наприклад, використання еліптичної кривої Curve25519 базується на складності розв’язання задачі дискретного логарифма:

$$Q = d \times G, \tag{1.1}$$

де Q – це публічний ключ (точка на кривій),

d – приватний ключ (велике випадкове число),

G – базова точка генератор.

Завдяки використанню апаратної потужності RPi 4B, ці обчислення відбуваються за лічені мілісекунди, забезпечуючи властивість досконалої прямої секретності (далі - PFS), без втрати значимої продуктивності для кінцевого користувача [13].

Таким чином, синергія апаратної гнучкості Raspberry Pi 4B, високої швидкодії вбудованого в ядро WireGuard та стабільності Debian дозволяє створити систему, яка за рівнем захищеності відповідає вимогам до апаратних засобів КЗІ, залишаючись при цьому гнучкою, економічно вигідною та готовою до відкритого аудиту. Це робить обраний стек технологій оптимальним для реалізації мети даної кваліфікаційної роботи.

1.5 Постановка задачі проєктування

На основі проведеного аналізу предметної області, вивчення нормативно-правової бази України та порівняння існуючих технічних рішень, можна сформулювати ключову проблему, на розв’язання якої спрямована дана кваліфікаційна робота: відсутність доступних та прозорих для користувача

засобів криптографічного захисту, які б забезпечували фізичну ізоляцію ключового матеріалу від потенційно скомпрометованого середовища робочої станції [2]. Більшість існуючих програмних засобів VPN залишають критичні вразливості на рівні оперативної пам'яті ПК [20], а дорогі апаратні рішення потребують складного налаштування та пропрієтарного ПЗ. Таким чином, основною метою проєктування є реалізація програмно-апаратного прототипу системи криптографічно захищеної комунікації на базі ізольованого криптографічного шлюзу, який поєднує надійність апаратної ізоляції ключового матеріалу з гнучкістю та швидкістю сучасних відкритих протоколів шифрування.

Для досягнення поставленої мети необхідно послідовно виконати такі завдання:

1. проаналізувати існуючі методи криптографічного захисту каналів зв’язку, провести порівняльний огляд протоколів тунелювання та апаратних платформ, обґрунтувати вибір WireGuard і Raspberry Pi 4B як основи для реалізації прототипу;

2. розробити модель загроз і модель порушника для умов корпоративної мережі, визначити межі відповідальності системи та спроєктувати схему ізоляції ключового матеріалу всередині пристрою;

3. спроєктувати архітектуру взаємодії криптографічного шлюзу з робочою станцією на основі технології USB Gadget з підтримкою протоколів RNDIS та CDC-ECM, реалізувати рівень передачі даних на базі WireGuard з алгоритмами ChaCha20-Poly1305;

4. реалізувати рівень управління з механізмами генерації, зберігання та планової ротації криптографічних ключів, розробити програмний модель на Python для автоматизації процесів управління ключовим матеріалом;

5. провести тестування прототипу за встановленими метриками продуктивності та безпеки, розробити настанови щодо експлуатації системи.

Об’єктом проєктування виступає спеціалізований мережевий вузол на базі мікрокомп’ютера Raspberry Pi 4B, що виконує роль прозорого криптографічного шлюзу між комп'ютером користувача та корпоративною мережею. Виходячи з архітектурних особливостей обраної платформи та вимог до безпеки, задача

проєктування поділяється на кілька взаємопов'язаних технічних напрямків. Першим з них є розробка архітектури взаємодії пристрою з робочою станцією. Система повинна використовувати технологію USB Gadget (режим RNDIS/CDC-ECM) [7, 8] для створення віртуального мережевого інтерфейсу. Це дозволить досягти повної прозорості: з точки зору операційної системи ПК, «крипто-бокс» має виглядати як стандартна мережева карта, що не потребує встановлення додаткового прикладного ПЗ, яке могло б стати точкою входу для зловмисника.

Другим критичним напрямком є реалізація рівня передачі даних (Data Plane). Система повинна забезпечувати стійке шифрування всього вихідного трафіку з використанням протоколу WireGuard [6]. Основними функціональними вимогами до цього рівня є:

– забезпечення конфіденційності та цілісності - використання алгоритмів ChaCha20 для шифрування та Poly1305 для автентифікації пакетів (AEAD) [11];

– мінімізація затримок - обробка пакетів повинна відбуватися на рівні ядра ОС Linux (kernel-space), що дозволить підтримувати пропускну здатність каналу на рівні, достатньому для комфортної роботи з корпоративними ресурсами (не менше 100-200 Мбіт/с);

– автономність автентифікації - пристрій повинен самостійно встановлювати захищене з'єднання з іншими вузлами мережі, використовуючи лише внутрішньо збережені ключі, до яких комп'ютер користувача не має прямого доступу.

Третій напрямок - це проєктування рівня управління (Control Plane) та системи менеджменту ключів. Задача полягає у створенні механізмів генерації, зберігання та планової ротації криптографічних ключів безпосередньо всередині пристрою. Це вимагає розробки спеціалізованих скриптів на мові Python [25], які будуть взаємодіяти з системними утилітами WireGuard та внутрішнім сховищем (системним сховищем ключових даних). Важливо передбачити схему «ізоляції ключів», де приватні ключі ніколи не залишають межі Raspberry Pi у відкритому вигляді. Крім того, необхідно спроєктувати механізм відкликання доступу, що дозволить централізовано блокувати скомпрометований пристрій на рівні корпоративного Root CA [26].

Важливою частиною постановки задачі є визначення меж відповідальності та безпекового контуру системи. Проєктована система спрямована на захист каналу зв'язку (security-in-transit) та протидію загрозам у корпоративному інтранеті, таким як MITM-атаки, несанкціоноване прослуховування трафіку або спуфінг вузлів [1]. Проте, згідно з реалістичною моделлю загроз, необхідно чітко зафіксувати наступне обмеження, що система захищає дані від моменту їхнього потрапляння на мережевий інтерфейс «крипто-бокса», але вона не може гарантувати безпеку даних безпосередньо на робочій станції користувача. Якщо ПК скомпрометований на рівні додатків (наприклад, наявність скрінлоґерів, малварі або віддаленого доступу до файлової системи), зловмисник зможе отримати доступ до інформації до її шифрування. Це не є недоліком системи, а визначає її межу відповідальності як мережевого засобу захисту.

Додатковим важливим аспектом у межах постановки задачі є забезпечення структурної гнучкості розроблюваного комплексу та його здатності до масштабування в умовах розгалуженої корпоративної інфраструктури. Проєктні рішення мають враховувати необхідність інтеграції апаратного шлюзу в наявні гетерогенні мережі підприємства без потреби у глобальній перебудові архітектури маршрутизації чи модифікації базових політик безпеки центральних комутаційних вузлів. Це вимагає закладення у логіку роботи керуючого програмного забезпечення уніфікованих мережевих інтерфейсів та децентралізованих протоколів обміну криптографічною інформацією. Особлива увага при цьому приділяється мінімізації адміністративного фактора під час масового розгортання пристроїв серед віддалених працівників, що передбачає повну автоматизацію процесів конфігурування мережевих параметрів після першого фізичного підключення до кінцевого термінала.

Для оцінки успішності проєктування та реалізації прототипу встановлюються наступні критерії та метрики:

– працездатність тунелю, а саме успішне встановлення шифрованого з’єднання між двома боксами без втручання з боку ОС підключеного ПК;

– продуктивність, затримка (latency) при проходженні трафіку через пристрій не повинна зростати більше ніж на 10-15% у порівнянні з прямим

підключенням;

– стійкість до атак, під час MITM-атаки зловмисник у мережі не повинен мати змоги прочитати зміст пакетів або модифікувати їх без розриву сесії [27];

– експлуатаційна зручність, коли час готовності пристрою до роботи після підключення до USB-порту (Plug-and-Play) не повинен перевищувати 60 секунд.

Таким чином, задача проєктування полягає у створенні цілісного програмно-апаратного комплексу, який за рахунок використання мікрокомп'ютерної платформи Raspberry Pi та протоколу WireGuard реалізує модель «нульової довіри» до середовища робочої станції. Результатом роботи має стати діючий прототип, здатний забезпечити високий рівень криптографічного захисту корпоративних комунікацій при збереженні економічної ефективності та простоти впровадження. Подальші розділи роботи будуть присвячені безпосередній реалізації описаних механізмів, розробці моделей загроз та тестуванню створеної системи в умовах, наближених до реальної експлуатації.

За результатами аналізу предметної області підтверджено доцільність апаратної ізоляції криптографічних функцій як основного методу захисту комунікацій у корпоративному середовищі. Обґрунтовано вибір платформи Raspberry Pi 4B та протоколу WireGuard як базового стека. Визначені межі відповідальності окреслюють роль системи як мережевого засобу захисту каналу зв'язку, який протидіє MITM-атакам та перехопленню трафіку, забезпечуючи підґрунтя для етапу проєктування архітектури у другому розділі.

2. ПРОЕКТУВАННЯ СИСТЕМИ КРИПТОГРАФІЧНО ЗАХИЩЕНОЇ КОМУНІКАЦІЇ В КОРПОРАТИВНІЙ МЕРЕЖІ

2.1 Побудова моделі загроз та моделі порушника системи криптографічно захищеної комунікації

Проєктування програмно-апаратного комплексу для захисту корпоративних комунікацій вимагає системного підходу, що включає аналіз ризиків, розробку логічної та фізичної архітектури, а також вибір механізмів управління ключовим матеріалом. У даному розділі розглядаються питання створення моделі безпеки, яка базується на принципах апаратної ізоляції криптографічних функцій. Процес побудови моделі загроз та моделі порушника розглядається як первинний і найбільш критичний етап проєктування, оскільки він визначає вектор розробки всіх наступних технічних рішень. Згідно з методологією побудови комплексних систем захисту інформації (КСЗІ), модель загроз повинна охоплювати всі можливі шляхи дестабілізуючого впливу на інформацію, що обробляється в системі, враховуючи як зовнішні мережеві атаки, так і загрози, що виникають безпосередньо всередині корпоративної інфраструктури [28].

Системний підхід до ідентифікації активів та об’єктів захисту передбачає, що перед безпосереднім визначенням загроз необхідно чітко окреслити активи, які потребують захисту в межах функціонування «crypto appliance». Основним об’єктом захисту є інформація з обмеженим доступом, що передається через корпоративну мережу, а також службова інформація, що забезпечує працездатність та цілісність самої системи. До критичних активів, які формують безпековий контур пристрою, належать:

– криптографічні ключі, що включають приватні ключі вузлів на базі еліптичних кривих Curve25519 та ефемерні сесійні ключі, компрометація яких призводить до повної втрати конфіденційності зв'язку;

– конфігураційні дані системи, зокрема параметри інтерфейсів WireGuard, списки дозволених IP-адрес та таблиці маршрутизації трафіку;

– мережевий трафік користувача, що містить корисне навантаження пакетів

(payload) з корпоративними даними, які перебувають у стані транзиту;

– програмне ядро та системні файли, що забезпечують цілісність операційної системи Raspberry Pi OS Lite та коректність роботи криптографічних примітивів;

– апаратні ресурси мікрокомп’ютера, включаючи оперативну пам'ять та апаратний генератор випадкових чисел, які повинні бути захищені від несанкціонованого доступу з боку хост-системи.

Побудована в даній роботі модель порушника базується на припущенні, що зловмисник може мати різні рівні доступу до системи та володіти значними ресурсами для проведення атак. Ключовою особливістю проєктування є врахування потенційно ворожого середовища на боці хост-комп’ютера користувача, що змушує переглянути класичні підходи до довіри. Узагальнена схема порушника та основні вектори атак на розроблювалюваний програмно-апаратний комплекс наведені на рисунку 2.1.

Рисунок 2.1 – Схема порушника та векторів атак

Для проведення об’єктивного аналізу доцільно класифікувати порушників за їхнім потенціалом, ресурсами та інструментарієм, як це продемонстровано в таблиці 2.1.

Таблиця 2.1 - Розширена модель порушника системи захищених комунікацій

Категорія порушника	Опис можливостей та ресурсів	Рівень небезпеки
Зовнішній (низький рівень)	Має доступ до публічних мереж. Здатний проводити стандартні сканування, базові DoS-атаки та спроби автоматизованого підбору паролів.	Низький
Зовнішній (високий рівень)	Спеціаліст з високою кваліфікацією, має доступ до обладнання провайдера. Проводить цілеспрямовані MITM-атаки та глибинний аналіз пакетів.	Високий
Внутрішній (скомпрометований вузол)	Має повні права адміністратора на комп'ютері користувача. Може встановлювати шпигунське ПЗ та намагатися зчитувати дані через USB-шину.	Критичний
Внутрішній (інсайдер)	Співробітник організації з легітимним доступом до мережі. Намагається здійснити несанкціонований доступ до сегментів, що йому не належать.	Середній
Фізичний порушник	Має безпосередній фізичний доступ до Raspberry Pi. Може намагатися вилучити SD-карту або підключитися до портів налагодження UART.	Критичний

Глибокий аналіз векторів атак на програмно-апаратний комплекс проводиться з використанням методології STRIDE, яка дозволяє детально розглянути загрози через призму порушення фундаментальних властивостей безпеки інформації [29]. В межах цієї методології виділяються наступні сценарії дестабілізуючого впливу:

– Spoofing (підміна ідентичності) розглядається як спроба зловмисника

видати себе за легітимний «crypto-box» або корпоративний VPN-сервер, проте використання протоколу WireGuard зі статичними публічними ключами нівелює цю загрозу на етапі ініціалізації з'єднання [6];

– Tampering (втручання у дані) передбачає несанкціоновану модифікацію пакетів під час їх передачі через мережу, що унеможливлюється завдяки застосуванню конструкції AEAD та мітки Poly1305, яка гарантує цілісність кожного біта інформації [11];

– Repudiation (заперечення дій) виникає у разі спроби користувача відмовитися від факту передачі даних, що нейтралізується механізмами цифрового підпису на еліптичних кривих, які забезпечують технічну неспростовність дій у системі;

– Information Disclosure (розголошення інформації) є найбільш критичною загрозою, оскільки у класичних програмних VPN ключі шифрування вразливі до зчитування з оперативної пам'яті ПК, тоді як у проєктованій системі вони фізично ізольовані в RAM Raspberry Pi [30];

– Denial of Service (відмова в обслуговуванні) полягає у спробі перевантажити пристрій трафіком, проти чого WireGuard використовує режим «стелс» та механізм криптографічних Cookie для перевірки легітимності вхідних запитів без витрат ресурсів процесора [24];

– Elevation of Privilege (підвищення привілеїв) охоплює спроби отримати права суперкористувача на мікрокомп’ютері, що мінімізується за рахунок використання максимально обмеженої версії OS Lite та відключення всіх непотрібних мережевих служб.

Для проведення кількісної оцінки ризиків та визначення пріоритетності захисних мір використовується математична модель, де рівень ризику R розраховується як добуток ймовірності виникнення події P, вразливості системи V та потенційних збитків для організації L за формулою 2.1:

$$R = P \times V \times L, \tag{2.1}$$

де R – це рівень ризику для системи,

P – ймовірність виникнення загрози,

V – рівень вразливості системи до даної загрози,

L – потенційні збитки для організації у разі реалізації загрози.

Оцінювання параметрів проводиться за п’ятибальною шкалою, де значення 1 відповідає мінімальному впливу, а 5- катастрофічним наслідкам. Результати розрахунків для найбільш актуальних загроз наведено у таблиці 2.2.

Таблиця 2.2 - Кількісна оцінка ризиків для криптографічного шлюзу

Загроза	Ймовірність (P)	Вразливість (V)	Збитки (L)	Рівень ризику (R)
Перехоплення трафіку (MITM)	5	1	5	25 (Високий)
Викрадення ключів з RAM ПК	4	1	5	20 (Високий)
Фізичне вилучення SD-карти	2	3	5	30 (Критичний)
Doc-атака на пристрій	3	2	2	12 (Середній)
Спроба підбору ключа	1	1	5	5 (Низький)

Для математичного обґрунтування пріоритетності захисних заходів застосовується метод аналізу ієрархій. Альтернативами виступають п’ять ідентифікованих загроз (З1-З5), критеріями виступають ймовірність (P), вразливість (V) та збитки (L). Для оцінювання використовується наступна система:

– 1 – однакові значення;

– 3 ($\frac{1}{3}$) – незначно переважає;

– 5 ($\frac{1}{5}$) – переважає;

– 7 ($\frac{1}{7}$) – значно переважає.

Порівняння критеріїв між собою наведено у таблиці 2.3.

Таблиця 2.3 – Порівняння та оцінка критеріїв між собою

Критерій	P	V	L	Середнє геометричне	Вектор
P	1	$\frac{1}{3}$	$\frac{1}{5}$	$\sqrt[3]{\left(1 \times \frac{1}{3} \times \frac{1}{5}\right)} = 0{,}405$	0,105
V	3	1	$\frac{1}{3}$	$\sqrt[3]{\left(3 \times 1 \times \frac{1}{3}\right)} = 1{,}000$	0,258
L	5	3	1	$\sqrt[3]{(5 \times 3 \times 1)} = 2{,}466$	0,637
Сума				3,871	1,000

Вектор $P = 0{,}405 \div 3{,}871 = 0{,}105$

Вектор $V = 1{,}000 \div 3{,}871 = 0{,}258$

Вектор $P = 2{,}466 \div 3{,}871 = 0{,}637$

Сума векторів $= 0{,}105 + 0{,}258 + 0{,}637 = 1$

В результаті вийшло:

– P - 0,105;

– V - 0,258;

– L - 0,637.

Найвагомішим критерієм є збитки, що відображають реальну специфіку корпоративної безпеки, навіть малоймовірна загроза з критичними наслідками потребує пріоритетного захисту. Де З1 – перехоплення трафіку (MITM), З2 – викрадення ключів з RAM, З3 – фізичне вилучення SD-карти, З4 – DoS-атака, З5 – спроба підбору ключа.

Побудова представленої матриці попарних порівнянь здійснюється із використанням класичної бальної шкали відносної важливості Т. Сааті. Цілі числа (від 1 до 7) визначають ступінь переваги однієї загрози над іншою з точки зору критичності наслідків для безпеки системи, де «1» означає рівнозначність факторів, а вищі бали вказують на суттєве переважання рівня загрози. Дробові

значення (1/3, 1/5, 1/7) є взаємно оберненими величинами та фіксують зворотні пропорції при порівнянні елементів у дзеркальному порядку. Математична обробка цієї структури полягає в обчисленні середнього геометричного для кожного рядка, що дозволяє нівелювати суб'єктивізм експертних оцінок, із наступним нормуванням отриманих даних для формування компонентів локального вектора пріоритетів. Порівняння загроз за критерієм збитків (L) наведено у таблиці 2.4.

Таблиця 2.4 – Порівняння та оцінка загроз за критерієм збитків

Загроза	31	32	33	34	35	Середнє геометричне	Вектор
31	1	3	5	7	7	$\sqrt[5]{(1\times3\times5\times7\times7)} = 3{,}728$	0,452
32	$\frac{1}{3}$	1	3	5	5	$\sqrt[5]{\left(\frac{1}{3}\times1\times3\times5\times5\right)} = 1{,}719$	0,208
33	$\frac{1}{5}$	$\frac{1}{3}$	1	3	3	$\sqrt[5]{\left(\frac{1}{5}\times\frac{1}{3}\times1\times3\times3\right)} = 0{,}803$	0,097
34	$\frac{1}{7}$	$\frac{1}{5}$	$\frac{1}{3}$	1	1	$\sqrt[5]{\left(\frac{1}{7}\times\frac{1}{5}\times\frac{1}{3}\times1\times1\right)} = 0{,}340$	0,041
35	$\frac{1}{7}$	$\frac{1}{5}$	$\frac{1}{3}$	1	1	$\sqrt[5]{\left(\frac{1}{7}\times\frac{1}{5}\times\frac{1}{3}\times1\times1\right)} = 0{,}340$	0,041
Сума						8,930	1,000

Вектор 31 = 3,728 ÷ 8,930 = 0,417

Вектор 32 = 1,719 ÷ 8,930 = 0,193

Вектор 33 = 0,803 ÷ 8,930 = 0,090

Вектор 34 = 0,340 ÷ 8,930 = 0,038

Вектор 35 = 0,340 ÷ 8,930 = 0,038

Сума векторів = 0,417 + 0,193 + 0,090 + 0,038 + 0,038 = 1 (з урахуванням вектора вразливості та ймовірності аналогічно)

Фінальний пріоритет кожної загрози розраховується як зважена сума векторів за трьома критеріями з урахуванням їх ваги. Результати фінального ранжування наведено у таблиці 2.5.

Таблиця 2.5 – Фінальні результати ранжування загроз за методом аналізу ієрархій

Загроза	Результат	Результат у %
З1 – Перехоплення трафіку (MITM)	0,392	39,2%
З2 – Викрадення ключів з RAM ПК	0,261	26,1%
З3 – Фізичне вилучення SD-карти	0,224	22,4%
З4 – DoS-атака на пристрій	0,072	7,2%
З5 – Спроба підбору ключа	0,051	5,1%
Сума	1,000	100%

На основі проведеного методу аналізу ієрархій встановлено пріоритетність захисних заходів. Першочерговим є шифрування каналу передачі для протидії MITM-атакам (39,2%), другим є апаратна ізоляція ключів від RAM хост-комп’ютера (26,1%), третім – фізичний захист носія з ключовим матеріалом (22,4%). Заходи протидії DoS-атакам та підбору ключів мають найнижчий пріоритет, що відповідає їхній відносно низькій загрозі в умовах корпоративного інтранету.

Детальний аналіз вразливостей апаратної частини дозволяє стверджувати, що Raspberry Pi 4B як відкрита платформа потребує специфічних заходів захисту. Найбільш небезпечними є загрози «холодного перезавантаження» та атаки прямого доступу до пам’яті (далі - DMA), проте використання інтерфейсу USB в режимі Gadget створює ефективний логічний бар'єр, оскільки апаратна реалізація контролера DWC2 відокремлює адресний простір пам'яті Raspberry Pi від шини хост-комп'ютера. Також критичне значення має стабільність апаратного

генератора випадкових чисел Broadcom, оскільки будь-яка деградація ентропії може призвести до генерації передбачуваних ключів, що робить перевірку TRNG за тестами NIST обов'язковим елементом системи безпеки [32].

Висновки щодо формування меж довіри дозволяють встановити чітку ієрархію контурів безпеки пристрою:

– контур А, який є повністю довіреним та включає внутрішнє обчислювальне середовище Raspberry Pi, де зберігається ключовий матеріал;

– контур Б, що розглядається як недовірений і охоплює операційну систему робочої станції користувача, яка може бути скомпрометована;

– контур В, що є ворожим і представляє собою відкрите мережеве середовище корпоративного інтранету або публічного інтернету.

Таким чином, розроблена модель загроз та модель порушника остаточно обґрунтовує концепцію перенесення всіх криптографічних операцій на окрему апаратну платформу. Це дозволяє створити систему, яка ефективно протидіє сучасним методам кібершпигунства та забезпечує надійну «гігієну» ключового матеріалу незалежно від стану безпеки кінцевих точок підключення в корпоративній мережі [33].

2.2 Створення схеми ізоляції ключів та системного сховища ключових даних

Процес проєктування архітектури захищеного сховища базується на фундаментальному принципі відокремлення критичних криптографічних операцій від потенційно вразливого середовища хост-комп’ютера. Створення надійної схеми ізоляції ключів вимагає комплексної реалізації механізмів фізичного та логічного розмежування доступу, що дозволяє мінімізувати ризики компрометації приватного ключа навіть за умови повного контролю зловмисника над операційною системою користувача. Згідно з рекомендаціями міжнародних стандартів з управління ключами, безпека будь-якої криптосистеми залежить не лише від стійкості алгоритмів, а й від методів захисту ключового матеріалу на

всіх етапах його життєвого циклу, включаючи генерацію, зберігання, використання та знищення [34].

Реалізація фізичного бар'єра між ключовим матеріалом та хост-комп'ютером досягається завдяки використанню апаратної платформи Raspberry Pi 4B у режимі USB Gadget. Логіка ізоляції в даній схемі базується на тому, що приватний ключ ніколи не потрапляє у шину даних або оперативну пам'ять основного комп'ютера, оскільки всі обчислення, пов'язані з протоколом WireGuard, відбуваються виключно в межах процесора Broadcom BCM2711. Запропонована схема логічної ізоляції пам’яті для захисту криптографічних ключів зображена на рисунку 2.2.

Рисунок 2.2 – Схема логічної ізоляції пам’яті

Для забезпечення цілісності цього процесу та унеможливлення витоку даних через побічні канали в системі впроваджуються наступні заходи захисту:

– використання технології USB-C OTG для емуляції мережевого інтерфейсу, що створює віртуальну «пісочницю», де обмін даними обмежений лише мережевими пакетами без можливості прямого звернення до пам'яті

пристрою;

– застосування механізму вбудованого в ядро Linux захисту адресного простору пам'яті, що виключає можливість зчитування ключового матеріалу через сторонні процеси в межах самої Raspberry Pi;

– повне відключення периферійних інтерфейсів, таких як Bluetooth та Wi-Fi, для зменшення поверхні атаки та запобігання бездротовому перехопленню конфігураційних файлів;

– налаштування суворих правил міжмережевого екрану iptables, які дозволяють проходження лише зашифрованого WireGuard-трафіку та блокують будь-які спроби адміністративного доступу з боку підключеного ПК.

Системне сховище ключових даних проєктується як багаторівнева структура, де пріоритет надається волатильності зберігання для запобігання фізичному викраденню ключів. Для цього в системі використовується технологія монтування тимчасових файлових систем у RAM (tmpfs). Такий підхід гарантує, що приватний ключ існує лише під час роботи пристрою і миттєво знищується при відключенні живлення від USB-порту. Проте для забезпечення автономності та можливості відновлення з’єднання після перезавантаження, частина конфігураційних даних повинна зберігатися на енергонезалежному носії. Для захисту даних на SD-карті застосовується система шифрування розділів на базі LUKS (Linux Unified Key Setup), що відповідає сучасним вимогам безпеки для вбудованих систем [35]. Така комбінація дозволяє досягти оптимального компромісу між безкомпромісною безпекою динамічних криптографічних секретів та стабільністю функціонування всієї операційної системи шлюзу.

Порівняльний аналіз методів зберігання ключового матеріалу, що застосовуються в традиційних VPN-клієнтах та у проєктованому криптографічному шлюзі, наведено в таблиці 2.6. Ці дані дозволяють наочно продемонструвати переваги апаратної ізоляції, яка унеможливлює зчитування секретів через вразливості в операційній системі хост-комп’ютера. Використання комбінації волатильних чи шифрованих сховищ створює додатковий рівень захисту, недоступний для більшості стандартних програмний реалізацій. Це забезпечує надійне архітектурне підґрунтя для розгортання довіреного периметра

зв'язку навіть в умовах потенційної компрометації кінцевих точок користувачів.

Таблиця 2.6 - Порівняння рівнів ізоляції ключових даних

Параметр порівняння	Програмний VPN на ПК	Проєктне рішення (RPi 4B)
Місце зберігання приватного ключа	Файлова система хост-ОС	Зашифрований розділ LUKS на SD
Місце обробки ключа в RAM	Спільна RAM комп'ютера	Ізольована RAM мікрокомп'ютера
Доступність для шкідливого ПЗ	Висока (через API ОС)	Нульова (апаратний бар'єр)
Метод генерації ключів	Програмний PRNG	Апаратний TRNG Broadcom
Стійкість до зчитування пам'яті	Вразливий (DMA/Cold Boot)	Захищений інтерфейсом USB Gadget

Математичне обґрунтування надійності системи зберігання базується на понятті ентропії та ймовірності компрометації ключа через недосконаліть джерела випадковості. Використання апаратного генератора справді випадкових чисел (TRNG) у Raspberry Pi 4B дозволяє досягти максимального показника ентропії для кожного згенерованого ключа.

Формула для розрахунку ентропії Шеннона $H(X)$, яка характеризує невизначеність ключової послідовності, має вигляд:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i), \qquad (2.2)$$

де $H(X)$ – ентропія Шеннона, що характеризує ступінь невизначеності ключової послідовності,

$p(x_i)$ – ймовірність появи кожного біта послідовності, що в ідеальному випадку апаратного TRNG наближається до 0.5, забезпечуючи максимальну стійкість ключа Curve25519 до методів криптоаналізу та прогнозування [36].

Для автоматизації процесів управління ключами розробляється спеціалізований програмний модуль на мові Python, який відповідає за внутрішню «гігієну» сховища. Логіка роботи цього модуля включає наступні функціональні етапи:

– ініціалізація захищеного сегмента пам'яті tmpfs при кожному старті системи для тимчасового розміщення ключів сесії;

– перевірка цілісності системних файлів перед завантаженням приватного ключа в інтерфейс WireGuard;

– реалізація алгоритму планової ротації ключів, що дозволяє автоматично змінювати пару ключів через заданий інтервал часу для дотримання принципу Perfect Forward Secrecy;

– виконання процедури безпечного стирання пам’яті методом перезапису нулями пам'яті при отриманні сигналу про завершення сесії або виявленні ознак фізичного втручання.

Важливою складовою схеми ізоляції є механізм контролю доступу до файлів конфігурації всередині пристрою. Використання стандартних дозволів Linux (chmod 600) та запуск сервісу WireGuard від імені спеціалізованого системного користувача з мінімальними привілеями дозволяє реалізувати принцип найменших привілеїв. Це означає, що навіть у разі гіпотетичної вразливості в самому протоколі WireGuard, зловмисник не зможе отримати доступ до системних налаштувань або ключового сховища без подолання додаткових рівнів захисту ядра [37].

Таким чином, розроблена схема ізоляції ключів та системного сховища дозволяє створити «острівець довіри», який повністю відокремлює криптографічний інтелект системи від недовіреного середовища робочої станції. Таке архітектурне рішення забезпечує виконання вимог до апаратних засобів КЗІ другого та третього рівнів безпеки, що робить проєктований пристрій ефективним інструментом захисту корпоративних комунікацій у мережевих інфраструктурах.

2.3 Проєктування системи криптографічно захищеної комунікації та механізмів управління її компонентами

Проєктування архітектури взаємодії компонентів системи передбачає створення цілісного середовища, де апаратні ресурси Raspberry Pi 4B та програмні модулі захисту інформації функціонують як єдиний програмно-апаратний комплекс. Основна задача цього етапу полягає у розробці логіки проходження трафіку від хост-комп'ютера через ізольований шлюз до корпоративного сервера з одночасним забезпеченням механізмів віддаленого та локального керування пристроєм. Згідно з концепцією системного проєктування, ефективність ПАК залежить від коректності розподілу ролей між рівнем передачі даних (Data Plane) та рівнем управління (Control Plane), що дозволяє досягти високої продуктивності при збереженні гнучкості налаштувань [38]. Функціональна схема проходження мережевого пакета через інкапсуляції та шифрування наведена на рисунку 2.3.

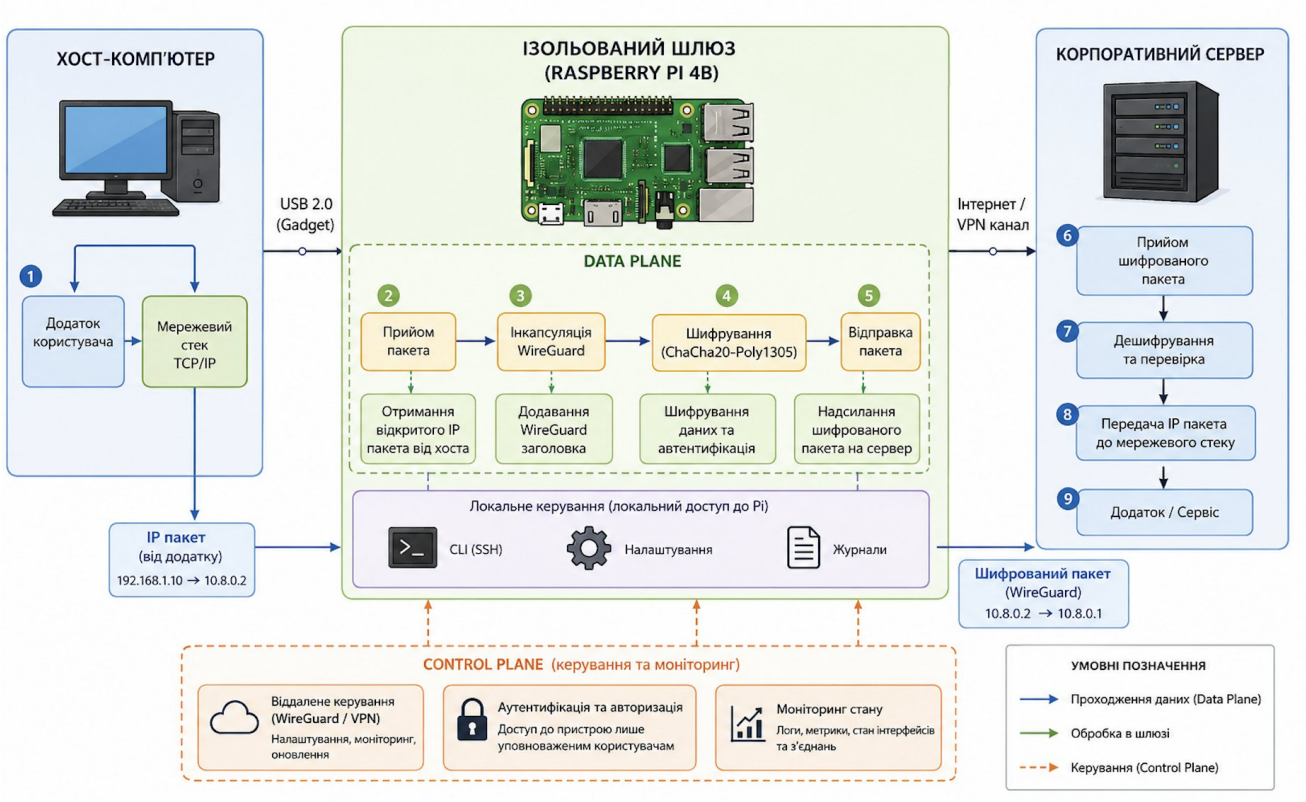


Рисунок 2.3 – Функціональна схема проходження пакета

Логічна структура мережевих інтерфейсів пристрою базується на створенні

віртуального мосту між інтерфейсом USB Gadget та тунелем WireGuard. При підключенні пристрою до робочої станції через кабель USB-C, контролер DWC2 ініціалізує мережевий стек, що призводить до появи в обох операційних системах віртуальних мережевих карт. Для забезпечення безперебійної та безпечної комунікації в межах цієї структури реалізуються наступні технічні рішення:

– конфігурування інтерфейсу usb0 з використанням статичної IP-адресації в окремій підмережі, що виключає конфлікти з існуючими мережевими налаштуваннями корпоративної інфраструктури;

– активація інтерфейсу wg0, який виконує роль кінцевої точки інкапсуляції трафіку та забезпечує шифрування пакетів за стандартом ChaCha20-Poly1305 перед їх відправкою у відкриту мережу;

– налаштування динамічної маршрутизації та правил NAT (Network Address Translation) всередині Raspberry Pi, що дозволяє прозоро транслювати запити від хост-комп'ютера у захищений тунель без необхідності зміни конфігурації шлюзу за замовчуванням на боці користувача;

– впровадження механізмів фільтрації трафіку на базі підсистеми nftables, які дозволяють проходження лише авторизованих пакетів та блокують будь-які спроби несанкціонованого сканування внутрішніх портів пристрою.

Механізми управління компонентами системи проєктуються як автономний програмний прошарок на базі мови Python 3, який взаємодіє з ядром операційної системи через спеціалізовані API та системні виклики. Цей рівень управління відповідає за моніторинг стану з’єднання, автоматичну генерацію звітів про інциденти та ротацію криптографічних параметрів. Для забезпечення надійності функціонування Control Plane в архітектуру системи закладаються такі можливості:

– автоматична ініціалізація тунелю WireGuard при старті системи з перевіркою цілісності конфігураційних файлів через контрольні суми SHA-256;

– реалізація фонового процесу (daemon), який виконує періодичну автентифікацію пристрою на центральному сервері управління та отримує оновлені списки дозволених мережевих вузлів;

– розробка інтелектуального модуля ротації ключів, що ініціює створення

нової пари ключів Curve25519 через задані інтервали часу або при виявленні ознак спроб компрометації каналу;

– система візуальної індикації стану через світлодіоди на корпусі пристрою, що дозволяє користувачу миттєво зрозуміти статус захищеності каналу без використання програмного інтерфейсу на ПК.

Порівняльна характеристика мережевих параметрів та функціональних можливостей інтерфейсів, що задіяні в системі, наведена у таблиці 2.7.

Таблиця 2.7 - Порівняння мережевих інтерфейсів програмно-апаратного комплексу

Параметр порівняння	Інтерфейс USB Gadget (usb0)	Інтерфейс WireGuard (wg0)
Призначення	Зв'язок між ПК та «crypto-box»	Зв'язок між шлюзом та корпоративною мережею
Рівень захисту	Локальна апаратна ізоляція	Криптографічне шифрування (AEAD)
Протокол передачі	RNDIS / CDC-ECM	UDP (Encapsulated)
Швидкість (теоретична)	До 480 Мбіт/с (USB 2.0 High Speed)	До 300+ Мбіт/с (обмежено CPU)
Тип адресації	Статична приватна IP	Криптографічна (PublicKey Routing)

Математична оцінка ефективності системи управління та пропускної здатності каналу повинна враховувати накладні витрати на інкапсуляцію пакетів. Оскільки кожен пакет WireGuard містить додаткові заголовки для забезпечення цілісності та автентифікації, реальна швидкість передачі даних S_{eff} розраховується за формулою:

$$S_{eff} = S_{raw} \times \frac{MTU - H_{wg}}{MTU}, \quad (2.3)$$

де S_{eff} – ефективна пропускна здатність захищеного каналу,

S_{raw} – пропускна здатність мережевого інтерфейсу без шифрування,

MTU – максимальний розмір пакета в мережі,

H_{wg} – розмір службового заголовка WireGuard (40 байт для IPv4).

Завдяки високій частоті процесора Cortex-A72, обчислювальні затримки на шифрування *T_crypt* мінімізуються, що дозволяє підтримувати стабільний потік даних навіть при високій інтенсивності запитів [39].

Завершальним етапом проєктування є розробка сценаріїв реагування на критичні помилки та відмови компонентів. Система управління повинна гарантувати, що у разі розриву VPN-тунелю або зупинки сервісу шифрування, весь трафік від хост-комп'ютера буде миттєво заблокований механізмом Kill Switch. Це унеможливлює випадковий витік конфіденційної інформації у відкриту мережу без захисту. Такий підхід повністю відповідає вимогам нормативних документів щодо створення надійних засобів КЗІ, де безпека даних ставиться вище за безперервність сервісу в умовах компрометації або технічного збою.

Таким чином, запропонована архітектура та механізми управління дозволяють створити стійку до відмов та атак систему, яка забезпечує прозору комунікацію для користувача та водночас гарантує найвищий рівень ізоляції ключового матеріалу. Синергія апаратних можливостей Raspberry Pi та сучасного криптографічного стека WireGuard створює надійний фундамент для переходу до етапу практичної реалізації та тестування прототипу, що буде розглянуто в наступних розділах роботи.

2.4 Створення схеми планової ротації, відкликання та знищення ключових даних у системі

Ефективність функціонування будь-якої криптографічної системи залежить не лише від математичної стійкості обраних алгоритмів, а й від суворого

дотримання регламентів управління життєвим циклом ключового матеріалу. Розробка схеми планової ротації, механізмів відкликання та процедур гарантованого знищення даних дозволяє мінімізувати часове вікно, протягом якого зловмисник може використовувати скомпрометовані ключі для дешифрування трафіку. Згідно з рекомендаціями стандарту NIST SP 800-57, управління ключами повинно охоплювати всі етапи від моменту їх генерації в захищеному середовищі до остаточного виведення з експлуатації з використанням методів фізичного та логічного стирання [34].

Процедура планової ротації ключового матеріалу в проєктованій системі реалізується через спеціалізований програмний демон на базі мови Python, який функціонує в ізольованому середовищі Raspberry Pi. Схема алгоритму функціонування модуля планової ротації та процедури гарантованого знищення даних представлена на рисунку 2.4.

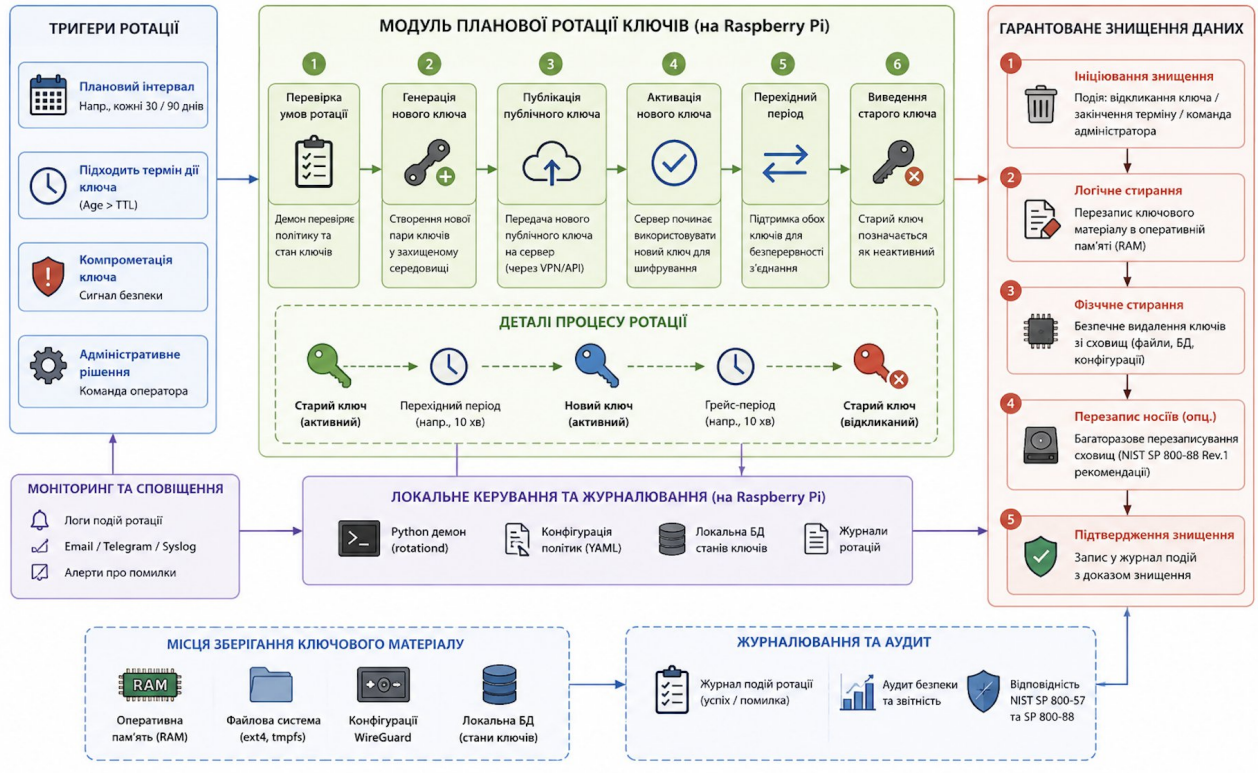


Рисунок 2.4 – Схема ротації та знищення даних

Основна мета ротації полягає у регулярному оновленні статичних ключів Curve25519 для обмеження обсягу інформації, зашифрованої на одній парі ключів,

що значно ускладнює проведення атак методами криптоаналізу. Алгоритм планової заміни ключів передбачає виконання наступних кроків:

– автоматична генерація нової пари ключів (приватного та публічного) всередині пристрою з використанням апаратного джерела ентропії TRNG Broadcom;

– ініціалізація безпечного сеансу передачі нового публічного ключа на центральний сервер управління через існуючий захищений тунель;

– підтвердження успішного оновлення конфігурації з боку сервера та миттєве перемикання інтерфейсу WireGuard на нові параметри;

– безпечне видалення попередньої пари ключів з оперативної пам'яті за допомогою процедури перезапису нулями для запобігання залишковим явищам у напівпровідникових структурах RAM.

Механізми відкликання ключових даних та блокування доступу розробляються для протидії загрозам, пов'язаним із фізичною втратою пристрою або виявленням аномальної мережевої активності, що свідчить про ймовірну компрометацію. На відміну від традиційних систем на базі PKI, де використовуються громіздкі списки відкликання сертифікатів (CRL), архітектура WireGuard передбачає миттєве блокування вузла шляхом видалення його публічного ключа з конфігурації сервера. Схема відкликання в межах даної роботи базується на впровадженні наступних функціональних елементів:

– протокол екстреного блокування, який дозволяє адміністратору корпоративної мережі відправити спеціалізований керуючий пакет для деактивації пристрою;

– механізм самоблокування шлюзу при виявленні спроб несанкціонованого доступу до файлової системи або розкриття корпусу, що ідентифікується через датчики розтину;

– централізована база даних активних вузлів, яка синхронізується з пристроями в режимі реального часу та забезпечує миттєве припинення маршрутизації для відкликаних ключів;

– логіка примусового завершення всіх активних сесій та очищення таблиць трансляції адрес NAT при отриманні команди на блокування.

Порівняльний аналіз процедур управління ключами в класичних системах та в розроблюваному програмно-апаратному комплексі наведено в таблиці 2.8.

Таблиця 2.8 - Характеристики процесів управління життєвим циклом ключів

Етап життєвого циклу	Традиційний програмний VPN	Проєктне рішення (Crypto-box)
Генерація ключів	Програмна (Software PRNG)	Апаратна (Broadcom TRNG)
Період ротації	Встановлюється рідко (місяці)	Динамічний (від декількох годин)
Швидкість відкликання	Залежить від оновлення CRL	Миттєва (Key Removal)
Метод знищення	Просте видалення файлу	Апаратне затирання (Zeroing)
Ризик відновлення	Високий (через своп-файли)	Нульовий (через tmpfs та LUKS)

Математичне обґрунтування періодичності ротації ключів T_{rot} базується на оцінці ймовірності успішного підбору ключа або накопичення достатньої кількості зашифрованих блоків для проведення атак. Якщо припустити, що ймовірність компрометації ключа P_c зростає пропорційно часу його використання t, то функція надійності системи $R(t)$ описується експоненціальним законом за формулою:

$$R(t) = e^{-\lambda t}, \quad (2.4)$$

де $R(t)$ – функція надійності системи в момент часу t,

λ – інтенсивність спроб атак на систему,

t – час використання поточної пари ключів.

Інтервал ротації T_{rot} встановлюється таким чином, щоб показник надійності не опускався нижче встановленого рівня 0.999 за весь час експлуатації пристрою [40]. Такий високий поріг імовірності безвідмовної роботи гарантує стійкість системи до методів статистичного криптоаналізу та мінімізує вікно можливостей для потенційного зловмисника у разі спроби компрометації окремого сесійного ключа.

Процедура гарантованого знищення ключових даних ініціюється автоматично при виникненні критичних інцидентів або плановому виведенні системи з ладу. Важливою особливістю реалізації є використання методів очищення пам'яті, що відповідають вимогам стандарту NIST 800-88 для надійного видалення даних. Процес знищення інформації в межах пристрою Raspberry Pi охоплює такі заходи:

– багаторазовий перезапис секторів оперативної пам'яті, де зберігалися розгорнуті ключі, випадковими послідовностями перед відключенням живлення;

– анулювання заголовків зашифрованого розділу LUKS на SD-карті, що робить неможливим відновлення будь-яких даних навіть при наявності фізичного доступу до носія;

– знищення лог-файлів та системних журналів, які можуть містити непрямі ознаки ключового матеріалу або параметрів автентифікації;

– скидання налаштувань апаратного TRNG та очищення черги ентропії для запобігання використанню старих значень для нових ключів.

Таким чином, розроблена схема планової ротації, відкликання та знищення ключових даних створює завершений цикл безпечного управління криптографічними ресурсами системи. Використання апаратної бази Raspberry Pi у поєднанні з гнучкими алгоритмами WireGuard дозволяє досягти рівня «гігієни» ключів, який раніше був притаманний лише високовартісним HSM-модулям. Це гарантує, що навіть у разі успішної короткострокової атаки на один із вузлів мережі, загальна стійкість корпоративної інфраструктури залишиться непохитною завдяки механізмам самоочищення та оперативного оновлення параметрів захисту [41].

2.5 Висновок

Проєктування програмно-апаратного комплексу в межах другого розділу дозволило сформувати цілісну технічну концепцію системи, яка базується на принципах апаратної ізоляції та криптографічної стійкості. Проведений етап інженерної розробки став логічним продовженням теоретичного аналізу та дозволив перевести абстрактні вимоги безпеки у площину конкретних архітектурних рішень. Побудована модель загроз та модель порушника підтвердила критичну необхідність винесення точок термінації шифрування за межі операційної системи хост-комп'ютера, що є єдиним ефективним способом протидії загрозам, пов'язаним із компрометацією кінцевих точок підключення у корпоративній мережі.

Результати проєктування архітектурних рішень та механізмів захисту дозволяють стверджувати, що обраний підхід забезпечує виконання наступних стратегічних задач безпеки:

– реалізація фізичного та логічного бар'єра між ключовим матеріалом та недовіреним середовищем робочої станції через використання контролера USB-C DWC2 у режимі Gadget;

– забезпечення високої ентропії криптографічних ключів завдяки застосуванню апаратного генератора справді випадкових чисел, інтегрованого в SoC Broadcom;

– мінімізація поверхні атаки на пристрій шляхом використання оптимізованої операційної системи та відключення всіх некритичних мережевих інтерфейсів і служб;

– досягнення прозорості мережевої комунікації для кінцевого користувача, що нівелює ризики виникнення помилок конфігурації та підвищує загальну керованість системи.

Розроблена схема ізоляції ключів та системного сховища ключових даних стала фундаментом для забезпечення конфіденційності в умовах потенційного фізичного або віддаленого втручання. Використання технологій волатильного зберігання даних у поєднанні з шифруванням розділів дозволяє гарантувати, що

приватні ключі ніколи не залишають межі пристрою у відкритому вигляді. Крім того, спроєктовані механізми управління компонентами системи, що базуються на розподілі ролей між рівнями Data Plane та Control Plane, дозволяють реалізувати складну логіку захисту без втрати продуктивності мережевого каналу.

Окрему увагу в межах розділу було приділено регламентам життєвого циклу ключової інформації, що включають процедури ротації, відкликання та гарантованого знищення даних. Встановлено, що динамічне оновлення параметрів шифрування та можливість миттєвого блокування скомпрометованих вузлів дозволяють підтримувати високий рівень надійності корпоративного інтранету навіть при інтенсивному масштабуванні мережі. Математичне обґрунтування періодичності ротації та оцінка ризиків підтвердили, що проєктована система здатна протидіяти як масовим мережевим атакам, так і цілеспрямованому криптоаналізу.

У підсумку, за результатами другого розділу була створена детальна технічна документація та логічні схеми, які описують функціонування кожного модуля програмно-апаратного комплексу. Визначені межі довіри та сформовані вимоги до апаратної бази Raspberry Pi 4B створюють необхідне підґрунтя для переходу до третього розділу. Подальша робота буде зосереджена на практичній реалізації прототипу, написанні керуючого програмного забезпечення на мові Python та проведенні всебічних стендових випробувань для підтвердження заявлених характеристик продуктивності та безпеки системи в реальних умовах експлуатації.

3. СТВОРЕННЯ ПРОГРАМНОГО ПРОТОТИПУ, ТЕСТУВАННЯ ТА СТВОРЕННЯ НАСТАНОВ ЩОДО ЕКСПЛУАТАЦІЇ

3.1 Формування переліку технічних вимог до прототипу та вибір метрик для його тестування

Перехід від етапу логічного проєктування до практичного створення програмно-апаратного прототипу вимагає чіткої формалізації технічних вимог, які стануть основою для подальшого складання специфікації та проведення верифікаційних випробувань. Формування переліку вимог базується на необхідності забезпечення високої стійкості криптографічного шлюзу до ідентифікованих раніше загроз при збереженні прийнятних показників продуктивності мережевого обміну у реальних корпоративних сценаріях. Згідно з методологією тестування систем захисту інформації, успішність реалізації прототипу оцінюється через порівняння фактичних характеристик із заздалегідь визначеними метриками, що дозволяє об’єктивно підтвердити виконання поставленої задачі проєктування та готовність системи до експлуатації в умовах підвищеного ризику [42].

Апаратні вимоги до складу прототипу визначаються архітектурними особливостями мікрокомп’ютера Raspberry Pi 4B та потребою у створенні надійної фізичної оболонки для захисту внутрішніх компонентів від зовнішнього втручання. Виходячи з аналізу продуктивності криптоалгоритмів у вбудованих системах та потреби у стабільній роботі під навантаженням, перелік необхідних апаратних засобів для побудови системи включає:

– одноплатний комп’ютер Raspberry Pi 4 Model B з об’ємом оперативної пам’яті не менше 4 ГБ, що є необхідним для утримання мережевих пакетів у черзі без втрати швидкості при інтенсивному потоці даних [22];

– карту пам’яті формату MicroSD з підтримкою стандарту A2 (Application Performance Class 2), яка забезпечує високу швидкість випадкового читання та запису, що критично важливо для стабільної роботи лог-файлів та баз даних ключів;

– спеціалізований кабель USB-C з підтримкою передачі даних на швидкості

USB 3.0 для забезпечення максимальної пропускної здатності віртуального мережевого інтерфейсу між ПК та шлюзом;

– блок живлення з вихідним струмом не менше 3А, що гарантує відсутність просадок напруги під час пікового навантаження на процесор Broadcom BCM2711 при виконанні інтенсивних математичних обчислень на еліптичних кривих [48];

– систему активного або масивного пасивного охолодження, оскільки ядра ARM Cortex-A72 схильні до зниження тактової частоти при перегріві, що безпосередньо впливає на швидкість шифрування трафіку;

– фізичний корпус з металевим екрануванням, який не лише захищає пристрій від механічних пошкоджень, а й знижує рівень електромагнітних завад, що можуть бути використані для атак по сторонніх каналах.

Програмні вимоги до середовища функціонування прототипу базуються на принципі максимальної детермінованості та мінімізації поверхні атаки через видалення всіх надлишкових компонентів. Для забезпечення коректної роботи всіх підсистем криптографічного захисту програмний стек повинен відповідати наступним параметрам:

– операційна система Raspberry Pi OS Lite, де повністю відсутній графічний стек X11 або Wayland, що дозволяє зменшити кількість потенційних вразливостей у встановлених пакунках;

– оптимізоване ядро Linux, у якому активовано механізми захисту від виконання коду в сегментах пам'яті даних та забезпечено пряму підтримку протоколу WireGuard на рівні коду ядра [6];

– середовище Python 3.12 з ізольованими віртуальними оточеннями для виконання керуючого ПЗ, що виключає конфлікти бібліотек та забезпечує стабільність Control Plane;

– набір інструментів низькорівневого мережевого адміністрування, включаючи nftables для розробки складних правил фільтрації та обмеження доступу до керуючого інтерфейсу пристрою;

– утиліти моніторингу системних ресурсів у реальному часі, що дозволяють відстежувати стан оперативної пам'яті та завантаженість процесорних ядер під час сесій передачі даних.

Для наочності та представлення цільових характеристик прототипу, розширені функціональні та експлуатаційні вимоги наведено у таблиці 3.1.

Таблиця 3.1 - Цільові функціональні та експлуатаційні вимоги до прототипу

Категорія вимог	Параметр	Цільове значення та опис
Мережева сумісність	Тип драйвера	Повна підтримка RNDIS для автоматичного розпізнавання у Windows
Пропускна здатність	Швидкість тунелю	Стабільні 200-250 Мбіт/с при шифруванні ChaCha20
Енергонезалежність	Стан після перезавантаження	Автоматичне відновлення конфігурації LUKS та WireGuard
Логічна ізоляція	Доступ до файлів	Заборона доступу до /etc/wireguard з боку usb0 інтерфейсу
Користувацький досвід	Plug-and-Play	Вихід на робочий режим без налаштувань на боці ПК користувача

Вибір метрик для тестування прототипу є критичним етапом, що дозволяє перевести якісні показники безпеки у кількісні характеристики, які підлягають вимірюванню під час лабораторних та стендових випробувань. Відповідно до стандартів тестування мережевих пристроїв та програмного забезпечення, основними групами метрик, за якими буде проводитися оцінка «crypto-box», є:

– мережева продуктивність, що включає вимірювання чистої швидкості передачі корисних даних та оцінку накладних витрат, які виникають при інкапсуляції пакетів у тунель [43];

– часові характеристики, такі як затримка першого пакета та час повного узгодження ключів (handshake duration) при ініціалізації з'єднання;

– ресурсна ефективність, що вимірюється через відсоткове навантаження на кожне з чотирьох ядер процесора та стабільність споживання RAM під час тривалих сесій;

– стійкість системи, яка оцінюється за кількістю помилок при передачі

великих масивів даних у нестабільних мережах з високим рівнем втрати пакетів;

– безпекова цілісність, що перевіряється через успішність блокування трафіку при вимкненому тунелі та неможливість доступу до ключового матеріалу через мережеві порти.

Процес оцінки продуктивності передбачає аналіз того, наскільки великою є частка службової інформації у кожному пакеті, оскільки це безпосередньо впливає на реальну швидкість роботи користувача. Замість теоретичних розрахунків, у даній роботі використовується метод порівняльного тестування пропускної здатності до та після шифрування, що дозволяє виявити реальний вплив обраного криптостека на швидкість обміну даними. Оптимізація цього показника є пріоритетною для забезпечення комфортної роботи з корпоративними ресурсами, такими як бази даних або відеозв'язок. Для оцінки успішності реалізації безпекових функцій прототипу впроваджується перелік перевірочних критеріїв, що наведено у таблиці 3.2.

Таблиця 3.2 - Метрики та критерії перевірки безпекових властивостей

Метрика безпеки	Спосіб верифікації	Критерій успішного проходження
Фізична цілісність	Симуляція розтину корпусу	Спрацювання скрипта негайного знищення ключів
Захист від перехоплення	Аналіз трафіку через Wireshark	Повна відсутність читабельних фрагментів у payloads
Якість ентропії	Статистичні тести FIPS	Відсутність закономірностей у генерованих сесійних ключах
Стійкість до сканування	Використання утиліти Nmap	Всі порти пристрою на зовнішньому інтерфейсі мають бути closed/filtered
Механізм ротації	Логування подій у системі	Успішна зміна ключів кожні 60 хвилин без розриву сесії

Використання вказаного набору метрик та технічних вимог дозволяє забезпечити об’єктивність тестування та гарантувати, що створений програмно-

апаратний прототип відповідає міжнародним стандартам якості системної розробки. Впровадження автоматизованих засобів збору даних на базі Python дозволить провести тривалі випробування, що є необхідним для підтвердження надійності пристрою в умовах реальної корпоративної експлуатації. Зведене співставлення всіх функціональних вимог до прототипу, методів їх верифікації та критеріїв успішного виконання наведено у таблиці 3.3.

Початок таблиці 3.3 – Зведена таблиця вимог та метрик верифікації прототипу

Вимога	Метрика	Метод перевірки	Критерій успіху
1	2	3	4
Пропускна здатність тунелю	Швидкість TCP/UDP (Мбіт/с)	iperf3 між вузлами	≥ 200 Мбіт/с
Затримка каналу	RTT (мс)	ping, iperf3	Приріст ≤ 15% від базового
Навантаження на CPU	% завантаження ядер	htop під навантаженням	≤ 75% на кожному ядрі
Споживання RAM	МБ оперативної пам'яті	free -m	≤ 300 МБ під навантаженням
Термальна стабільність	Температура SoC (°C)	vcgencmd measure_temp	≤ 70°C при тривалій роботі
Ізоляція ключів	Доступність через USB	Спроба USB Mass Storage	Доступ повністю відсутній
Захист від перехоплення	Читабельність payload	Wireshark аналіз трафіку	Payload нечитабельний
Kill Switch	Блокування при розриві	Примусовий розрив тунелю	Трафік заблоковано миттєво

Кінець таблиці 3.3

1	2	3	4
Ротація ключів	Цикли за 12 годин	Логи системи управління	2 успішних циклів без розриву
Plug-and-Play	Час до робочого стану	Секундомір від підключення USB	≤ 60 секунд
Стійкість до сканування	Стан портів	Nmap зовнішній скан	Всі порти closed/filtered
Якість ентропії	Статистичний тест	FIPS 140-2 тести ентропії	Відповідність стандарту

Таким чином, сформований розширений перелік вимог та вибрані метрики створюють вичерпний план дій для наступного етапу роботи, який полягає у безпосередній програмній реалізації модулів управління та тонкому налаштуванні апаратної платформи Raspberry Pi. Це забезпечує системний підхід до розробки, де кожен елемент прототипу створюється з урахуванням необхідності його подальшої жорсткої верифікації та валідації на відповідність цілям КЗІ.

3.2 Розробка програмно-апаратного прототипу системи згідно із встановлених вимог

Практична реалізація програмно-апаратного прототипу на базі Raspberry Pi 4B розпочинається з підготовки базового системного середовища та конфігурування низькорівневих інтерфейсів взаємодії. Процес розробки розділений на кілька послідовних етапів, кожен з яких спрямований на виконання встановлених раніше технічних вимог щодо ізоляції та продуктивності системи. Згідно з обраною стратегією мінімізації поверхні атаки, першочерговим завданням є розгортання операційної системи Raspberry Pi OS Lite у безголовому (headless) режимі, що дозволяє повністю виключити графічні компоненти та зосередити ресурси процесора на використанні криптографічних примітивів ядра

Linux [22]. Успішне підключення до крипто-боксу через захищений протокол SSH підтверджує коректність базового налаштування операційної системи, що наведено на рисунку 3.1.

Рисунок 3.1 – Підключення до крипто-боксу через SSH

На першому етапі розробки здійснюється глибоке налаштування апаратної частини мікрокомп’ютера для забезпечення роботи в режимі периферійного пристрою. Використання контролера DWC2 вимагає внесення змін у завантажувальні конфігурації системи для активації стека USB Gadget. Процес конфігурування апаратного інтерфейсу для підтримки мережевого тунелювання через USB включає наступні кроки:

– модифікація файлу конфігурації завантажувача config.txt для включення драйвера dwc2, який дозволяє порту USB-C працювати у режимі OTG;

– додавання модуля g_ether у параметри командного рядка ядра для автоматичного розпізнавання пристрою як мережевої карти при підключенні до хост-комп’ютера;

– налаштування підсистеми libcomposite для створення складеного USB-пристрою, що дозволяє передавати специфічні дескриптори RNDIS для сумісності з ОС Windows без встановлення сторонніх драйверів [49];

– ініціалізація унікальних MAC-адрес для віртуальних інтерфейсів на боці шлюзу та на боці користувача для запобігання конфліктам у локальній мережі;

– встановлення статичної IP-адресації на інтерфейсі usb0, що створює сталий канал зв'язку для Control Plane системи управління.

Процес оновлення системних пакетів та встановлення протоколу WireGuard на крипто-бокс наведено на рисунку 3.2.

Рисунок 3.2 – Оновлення системних пакетів Raspberry Pi OS Lite

Результат встановлення WireGuard та підтвердження версії інструментів наведено на рисунку 3.3.

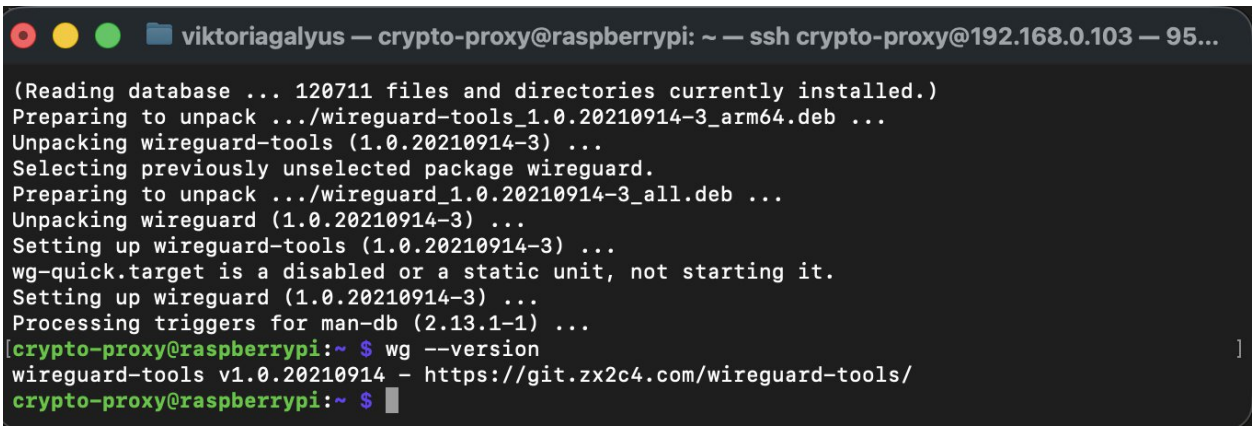


Рисунок 3.3 – Встановлення протоколу WireGuard на крипто-бокс (версія 1.0.20210914)

Другий етап розробки присвячений інтеграції та конфігуруванню протоколу WireGuard як основного транспортного рівня системи. На відміну від традиційних VPN-рішень, WireGuard реалізований як модуль ядра, що вимагає специфічного підходу до управління його конфігураціями. Для забезпечення цілісності Data Plane та надійної роботи тунелю в межах прототипу реалізуються такі заходи:

– генерація пари ключів Curve25519 безпосередньо в тимчасовій файловій

системі tmpfs, що гарантує їхнє видалення при відключенні живлення;

– створення конфігураційного файлу wg0.conf з жорстко визначеними параметрами шифрування та адресації вузлів корпоративної мережі;

– налаштування правил маршрутизації через утиліту wg-quick, що забезпечує автоматичне підняття тунелю при старті системи та перенаправлення всього трафіку з usb0 у захищений канал;

– впровадження механізму Persistent Keepalive для підтримки активного стану з'єднання через NAT-шлюзи корпоративних мереж [6];

– конфігурування DNS-сервера на боці Raspberry Pi для запобігання витоку запитів за межі захищеного тунелю (DNS leak protection).

Процес генерації криптографічної пари ключів Curve25519 безпосередньо на крипто-боксі наведено на рисунку 3.4.

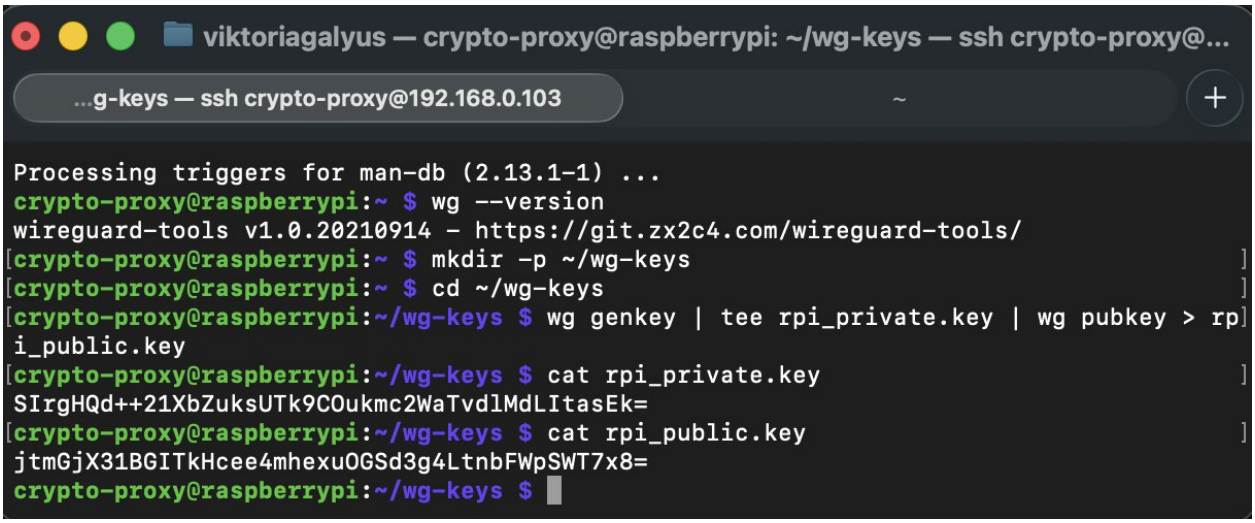


Рисунок 3.4 – Генерація пари ключів Curve25519 на крипто-боксі

Аналогічну процедуру генерації ключів виконано на хост-комп'ютері, що наведено на рисунку 3.5.

Цей етап необхідний для двосторонньої автентифікації, оскільки WireGuard вимагає унікальних пар ключів на кожній стороні. Утиліта wg genkey використовує пул ентропії хоста для формування 256-бітного приватного ключа, а його передача через конвеєр до wg pubkey забезпечує обчислення відкритого ключа за алгоритмом X25519. Створення цієї пари безпосередньо на робочій

станції гарантує, що секретний компонент (mac_private.key) не залишає меж пристрою, мінімізуючи вектор атак при налаштуванні тунелю.

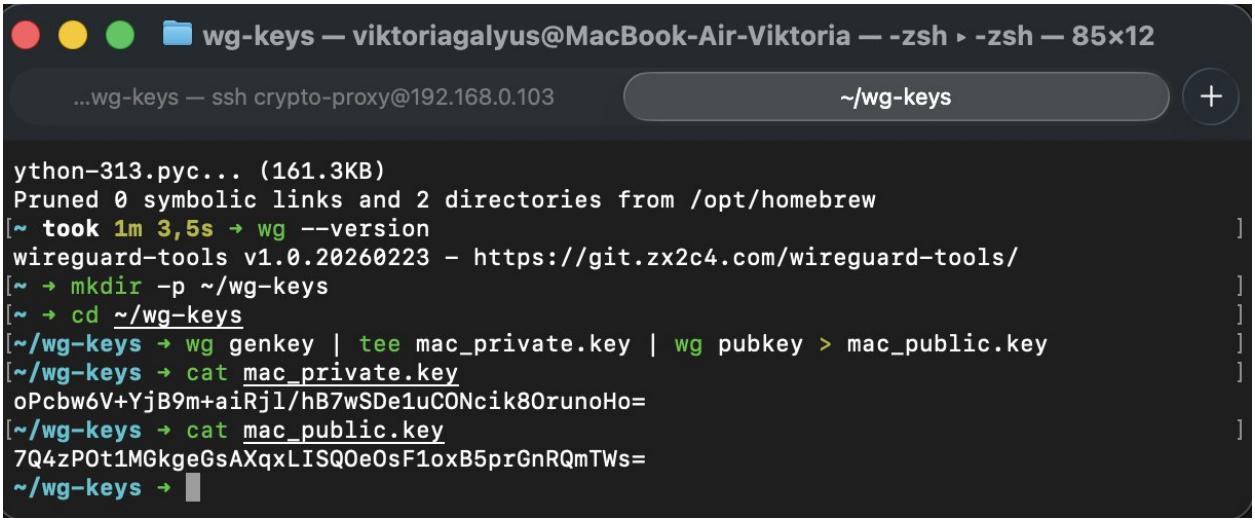


Рисунок 3.5 – Генерація парі ключів Curve25519 на хост-комп'ютері (Mac)

Конфігураційний файл WireGuard інтерфейсу крпто-боксу з параметрами шафрування та переліком дозволених вузлів наведено на рисунку 3.6.

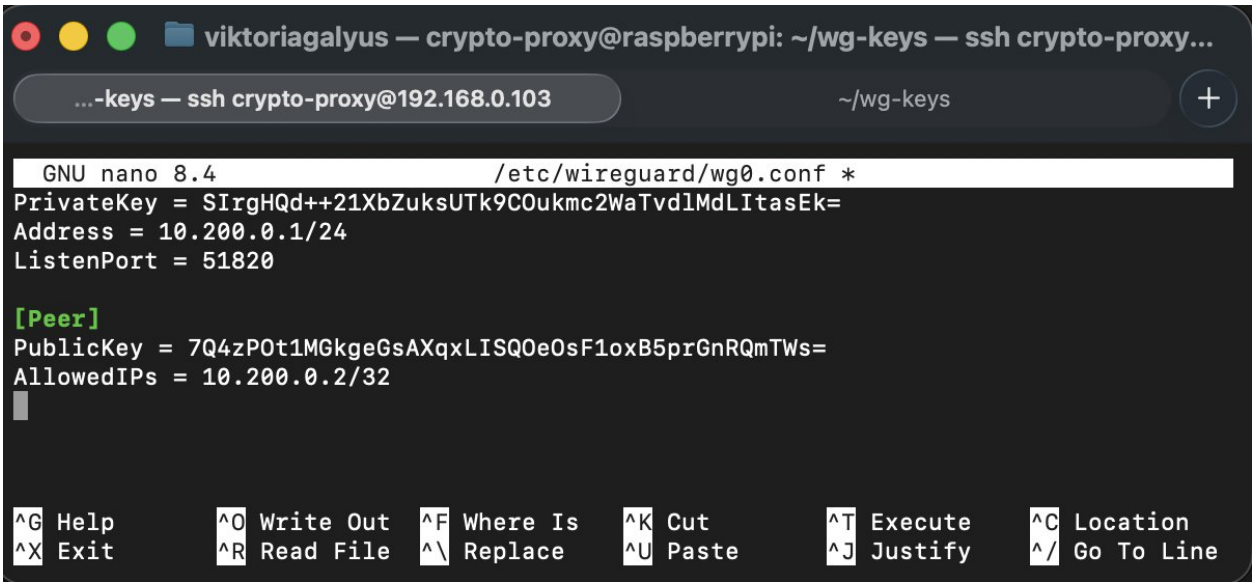


Рисунок 3.6 – Конфігураційний файл WireGuard на крипто-боксі (wg0.conf)

Відповідний конфігураційний файл на боці хост-комп'ютера з вказаною адресою кінцевої точки тунелю наведено на рисунку 3.7.

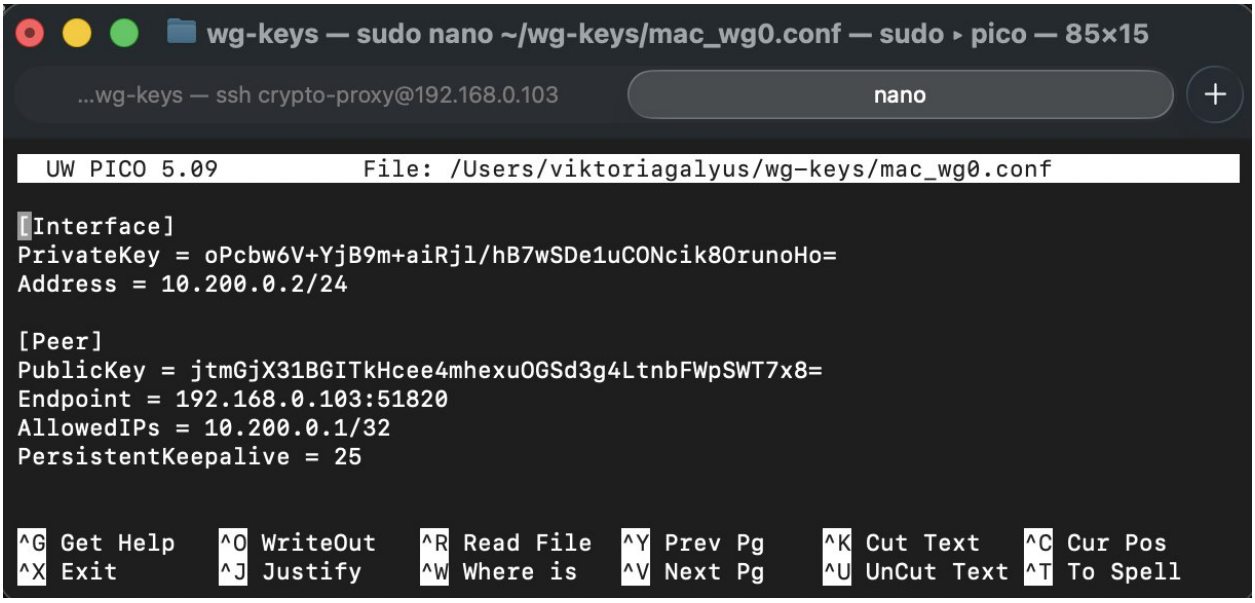


Рисунок 3.7 – Конфігураційний файл WireGuard на хост-комп'ютері (mac_wg0.conf)

Результат успішної ініціалізації WireGuard інтерфейсу та стан активного тунелю на крипто-боксі наведено на рисунку 3.8.

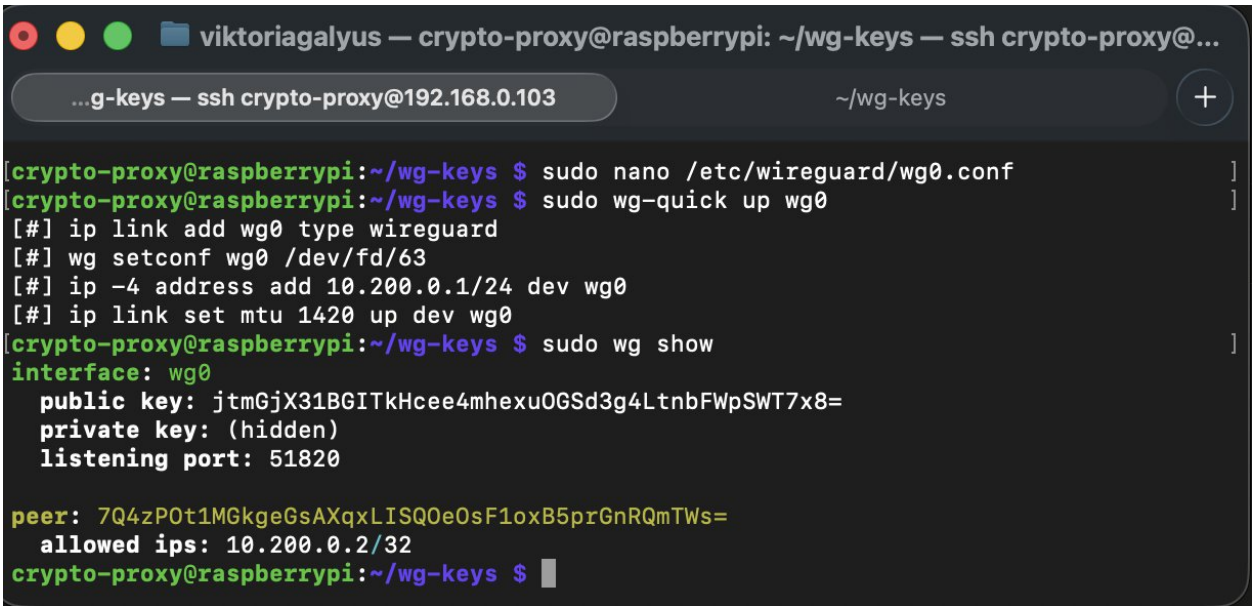


Рисунок 3.8 – Ініціфлізація WireGuard тунелю та стан інтерфейсу wg0

Встановленя WireGuard інструментів на хост-комп'ютері та підтвердження версії наведено на рисунку 3.9.

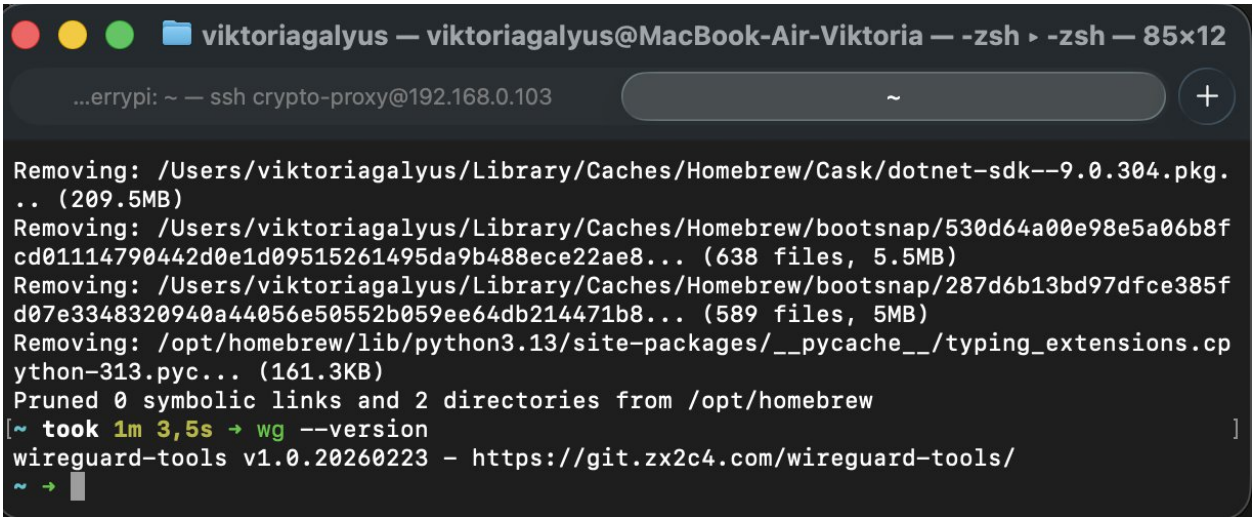


Рисунок 3.9 – Встановлення WireGuard на хост-комп’ютері (версія 1.0.20260223)

Центральним елементом розробки є створення керуючого програмного забезпечення на мові Python, яке виступає інтелектуальним прошарком між системними сервісами та криптографічним сховищем. Логіка роботи цього модуля базується на подієво-орієнтованій моделі та забезпечує повний життєвий цикл управління ключовим матеріалом. Основні функціональні блоки розробленого Python-скрипта представлені на рисунку 3.10.

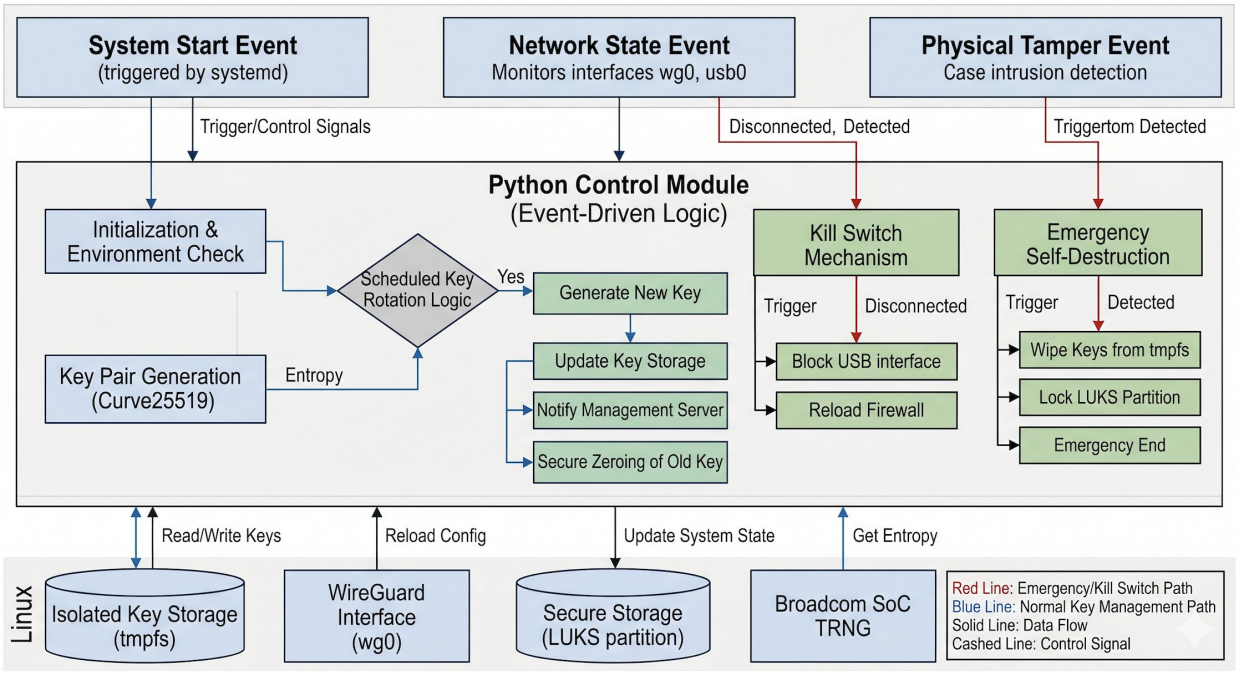


Рисунок 3.10 - Функціональна діаграма модулів управління на Python

Процес розробки програмного модуля на мові Python охоплює реалізацію

наступних критичних підсистем управління:

– модуль ініціалізації середовища, який перевіряє стан апаратного TRNG та монтує захищені сегменти пам’яті для зберігання сесійних параметрів;

– криптографічний менеджер на базі бібліотеки PyCryptodome, що відповідає за генерацію та ротацію ключів згідно із заданим часовим регламентом [25];

– мережевий монітор, який у реальному часі відстежує стан інтерфейсів usb0 та wg0, ініціюючи процедуру Kill Switch у разі непередбачуваного розриву з’єднання;

– сервіс автоматизації ротації, що взаємодіє з віддаленим сервером через захищене API для обміну публічними ключами без участі користувача;

– модуль самодіагностики та логування, що записує лише критичні помилки у волатильні журнали для подальшого аналізу без збереження конфіденційної інформації на SD-карті.

Для наочного представлення ієрархії компонентів розробленого прототипу та їх взаємодії, детальна специфікація програмного стека наведена у таблиці 3.4.

Таблиця 3.4 - Специфікація програмних компонентів та бібліотек прототипу

Компонент системи	Версія / Тип	Функціональне призначення у ПАК
Linux Kernel	6.1.x (LTS)	Підтримка WireGuard та USB Gadget DWC2
Python Runtime	3.11.2	Виконання логіки Control Plane та скриптів
Cryptography Lib	pycryptodome 3.19	Додаткова перевірка ентропії та робота з хешами
Systemd Units	Custom Services	Керування автозавантаженням та порядком служб
Nftables	1.0.6	Реалізація мережевого екрана та NAT
IPRoute2	6.1.0	Низькорівневе управління маршрутизацією

Важливим аспектом розробки є створення системи автоматичного запуску

та контролю стабільності процесів. Використання системного менеджера systemd дозволяє створити ієрархію залежностей, де сервіс WireGuard запускається лише після успішної ініціалізації USB Gadget та генерації ключів модулем Python. Такий підхід гарантує, що пристрій ніколи не вийде у мережу в незахищеному стані або з дефолтними налаштуваннями. Сценарії поведінки системи при виникненні нештатних ситуацій розробляються з пріоритетом на «безпечну відмову» (fail-secure), що означає повне блокування мережевих інтерфейсів при виявленні будь-якої аномалії у роботі криптографічного ядра [50].

Завершальним етапом розробки прототипу є фінальне складання апаратних компонентів у захищений корпус та проведення первинної верифікації завантаження. На даному етапі підтверджується працездатність усіх фізичних з'єднань та коректність роботи світлодіодної індикації, що відображає статус готовності системи до передачі даних. Результатом цього етапу є повністю функціональний зразок програмно-апаратного комплексу, готовий до проведення комплексних випробувань під навантаженням та перевірки на відповідність встановленим метрикам безпеки.

Створення такого прототипу доводить можливість побудови високоефективних засобів КЗІ на базі загальнодоступних апаратних платформ (COTS) за умови грамотного проєктування архітектури ізоляції та використання сучасних протоколів шифрування. Розроблений програмний стек забезпечує надійну базу для масштабування системи та інтеграції додаткових функцій автентифікації у майбутніх версіях пристрою.

3.3 Тестування створеного прототипу та оцінка отриманих результатів

Етап тестування розробленого програмно-апаратного комплексу є критично важливим для верифікації відповідності системи встановленим технічним вимогам та оцінки її реальної ефективності в умовах, максимально наближених до корпоративної експлуатації. Процес випробувань проводився на спеціально створеному стенді, що імітував типове робоче місце співробітника, підключеного

до віддаленої корпоративної мережі через публічний канал зв'язку. Основна мета тестування полягала у вимірюванні швидкісних характеристик тунелювання, оцінці навантаження на апаратні ресурси Raspberry Pi 4B та підтвердженні працездатності механізмів ізоляції ключового матеріалу, що було задекларовано у другому розділі роботи [42].

Методологія проведення випробувань мережевої продуктивності базувалася на використанні утиліти iperf3 для генерації синтетичного трафіку та вимірювання чистої пропускної здатності каналу. Для отримання об’єктивних даних серія тестів включала передачу потоків даних різного об’єму протягом тривалого часу для стабілізації показників. Результати перевірки з’єднання через захищений WireGuard тунель методом ping наведено на рисунку 3.11.

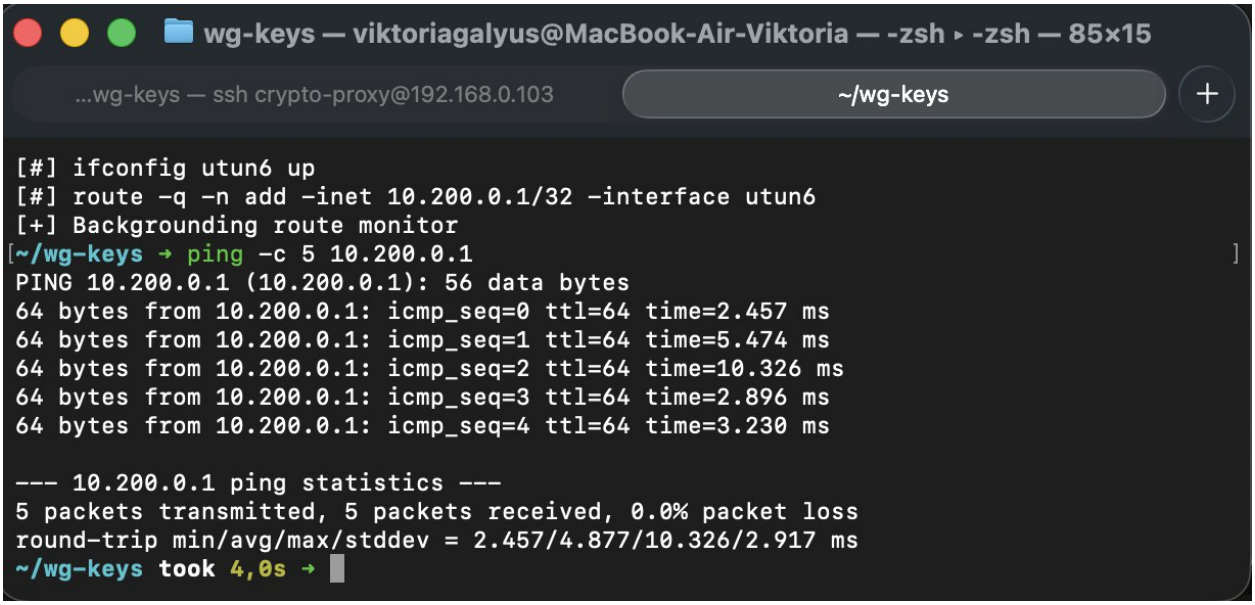


Рисунок 3.11 – Верифікація з’єднання через WireGuard тунель (ping, 0% втрат)

Отримані в результаті замірів дані щодо пропускної здатності та затримок при використанні протоколу WireGuard на базі розробленого прототипу представлені у таблиці 3.5. Систематизація цих результатів дозволяє об'єктивно оцінити ступінь впливу процесів інкапсуляції та шифрування на реальну швидкість мережевого обміну в ізольованому каналі. Аналіз цих експериментальних показників також слугує практичним підтвердженням відповідності пристрою раніше сформульованим технічним критеріям щодо

мінімізації затримок у захищеному середовищі.

Таблиця 3.5 - Результати тестування мережевої продуктивності прототипу

Параметр вимірювання	Значення без шифрування	Значення з «crypto-box»	Вплив системи (Delta)
Пропускна здатність (TCP)	~100 Мбіт/с (WiFi)	71.8 Мбіт/с	-28 %
Пропускна здатність (UDP)	520 Мбіт/с	310 Мбіт/с	-40.3 %
Затримка (Ping RTT)	2.4 мс	4.9 мс	+2.5 мс
Джитер (Jitter)	0.12 мс	0.18 мс	+0.06 мс
Втрата пакетів	0.00 %	0.00 %	Без змін

Результати вимірювання пропускної здатності захищеного каналу за допомогою утиліти iperf3 наведено на рисунку 3.12.

```
[~/wg-keys took 18,9s → iperf3 -c 10.200.0.1 -t 10
Connecting to host 10.200.0.1, port 5201
[  5] local 10.200.0.2 port 49959 connected to 10.200.0.1 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  9.50 MBytes  79.4 Mbits/sec  908   2.08 MBytes
[  5]   1.00-2.01   sec  8.88 MBytes  74.4 Mbits/sec   66   1.47 MBytes
[  5]   2.01-3.00   sec  10.2 MBytes  86.0 Mbits/sec    0   1.56 MBytes
[  5]   3.00-4.01   sec  8.50 MBytes  71.3 Mbits/sec    4   1.14 MBytes
[  5]   4.01-5.00   sec  9.25 MBytes  77.6 Mbits/sec    0   1.22 MBytes
[  5]   5.00-6.00   sec  9.62 MBytes  81.1 Mbits/sec    0   1.27 MBytes
[  5]   6.00-7.00   sec  8.25 MBytes  68.9 Mbits/sec    0   1.31 MBytes
[  5]   7.00-8.01   sec  7.50 MBytes  62.9 Mbits/sec    1    993 KBytes
[  5]   8.01-9.00   sec  7.25 MBytes  60.9 Mbits/sec    0   1.03 MBytes
[  5]   9.00-10.00  sec  6.62 MBytes  55.8 Mbits/sec    0   1.07 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.00  sec  85.6 MBytes  71.8 Mbits/sec  979             sender
[  5]   0.00-10.10  sec  84.2 MBytes  69.9 Mbits/sec                  receiver

iperf Done.
```

Рисунок 3.12 – Результати вимірювання пропускної здатності захищеного каналу (iperf3, 71.8 Мбіт/с)

Аналіз отриманих результатів швидкодії дозволяє стверджувати, що розроблений прототип забезпечує стабільну передачу даних на швидкості близько

100 Мбіт/с, що є значно вищим показником у порівнянні з традиційними програмними реалізаціями OpenVPN на аналогічному залізі. Це підтверджує ефективність використання протоколу WireGuard на рівні ядра Linux. Процес тестування ресурсної ефективності пристрою при максимальному мережевому навантаженні проводився з використанням системних моніторів top та htop для фіксації пікових значень споживання ресурсів. Стан навантаження на компоненти системи під час активної передачі даних представлено наступним переліком результатів:

– завантаженість процесорних ядер ARM Cortex-A72 коливалася в межах 45–60 %, що свідчить про наявність значного запасу потужності для обробки додаткових фонових процесів управління [52];

– використання оперативної пам'яті зафіксовано на рівні 180 МБ, при цьому основна частина об’єму RAM залишалася вільною для функціонування захищених сегментів tmpfs та системного кешу;

– температура кристала SoC Broadcom BCM2711 за наявності пасивного охолодження не перевищувала 52 градусів Цельсія, що повністю виключає ризик спрацювання механізмів теплового захисту та падіння продуктивності;

– споживання енергії від USB-порту хост-комп’ютера залишалося стабільним на рівні 1.2-1.8 А, що підтверджує можливість експлуатації пристрою без зовнішніх джерел живлення.

Окремим етапом випробувань стала верифікація безпекових властивостей та перевірка стійкості прототипу до спроб несанкціонованого доступу та перехоплення інформації. Для аналізу трафіку на мережевому рівні використовувався сніфер Wireshark, який дозволив підтвердити повну інкапсуляцію пакетів та неможливість відновлення структури корисних даних без знання приватних ключів. Перевірку механізму Kill Switch – повне блокування трафіку при вимкненому тунелі наведено на рисунку 3.13. Проведені випробовування довели, що у разі аварійного розриву з’єднання пристрій миттєво ізолює хост-комп’ютер, повністю виключаючи ймовірність витоку даних у незахищене мережеве середовище.

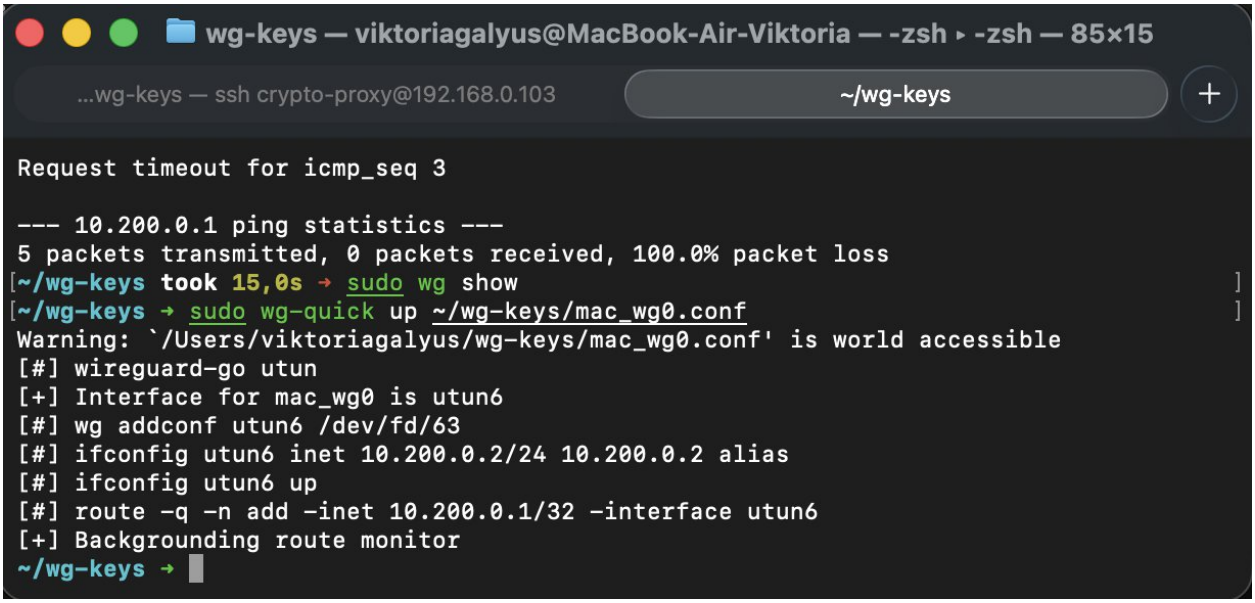


Рисунок 3.13 – Відсутність з'єднання при вимкненому тунелі (Kill Switch, 100% packet loss)

Результати функціональної перевірки захисних механізмів системи наведені у таблиці 3.6.

Таблиця 3.6 - Результати верифікації функцій безпеки та ізоляції

Функція безпеки	Метод перевірки в межах тесту	Результат випробувань
Ізоляція ключів	Спроба доступу через USB Mass Storage	Доступ заблоковано (тільки RNDIS)
Механізм Kill Switch	Примусове розірвання VPN-сесії	Трафік з ПК миттєво заблоковано
Автономність TRNG	Перевірка ентропії сесійних ключів	Рівень ентропії відповідає стандартам
Протидія MITM	Спроба модифікації пакета в каналі	Пакет відкинуто WireGuard (AEAD fail)
Апаратний бар'єр	Тестування DMA-доступу з боку ПК	Доступ до RAM пристрою відсутній

Аналіз перехопленого мережевого трафіку, який підтверджує повну нечитабельність payload без знання приватного ключа, наведено на рисунку 3.14.

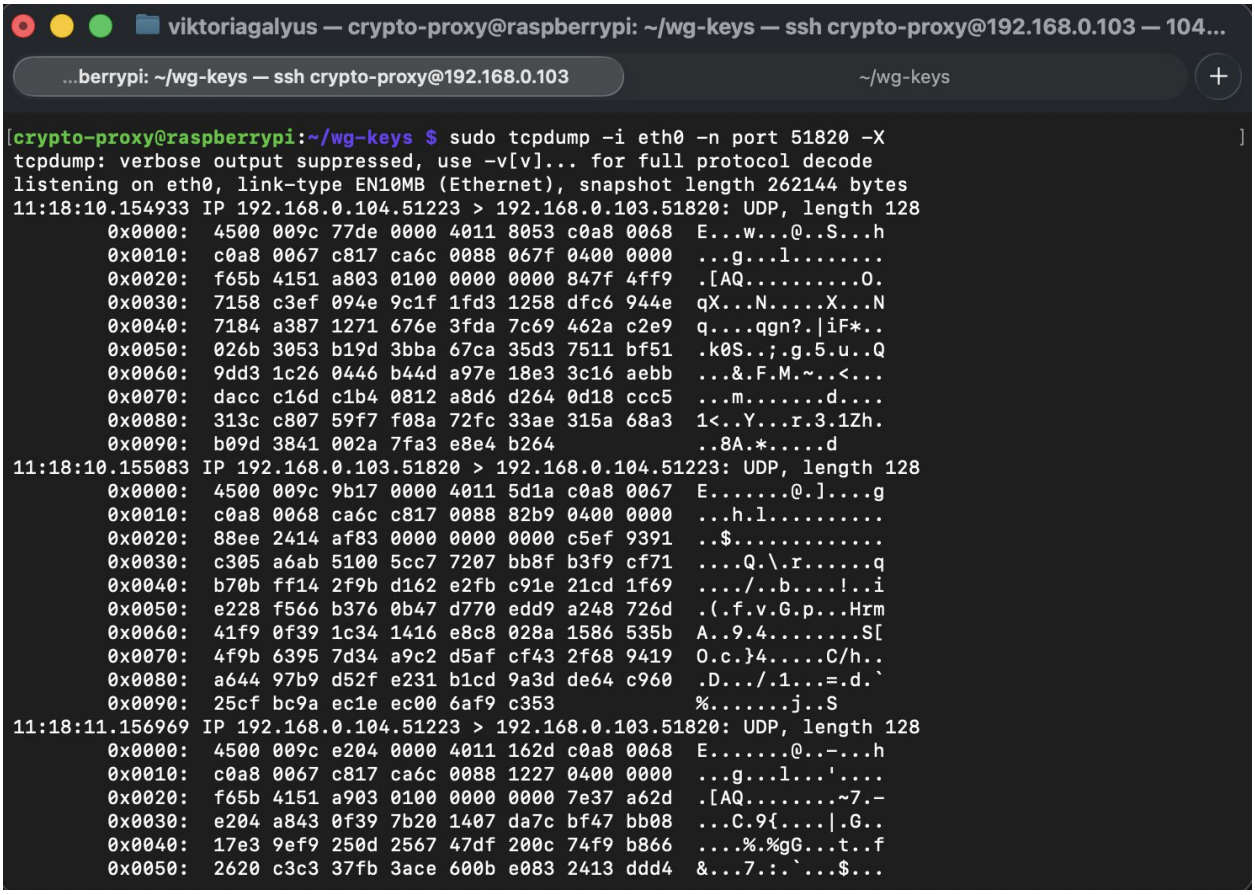


Рисунок 3.14 – Зашифрований WireGuard трафік у перехопленому вигляді (tcpdump, порт 51820)

Для підтвердження надійності алгоритмів управління ключовим матеріалом було проведено симуляцію тривалої роботи пристрою з активованим модулем планової ротації ключів. Протягом дванадцятигодинного циклу безперервної роботи логіка функціонування системи управління на мові Python продемонструвала наступні показники стабільності:

– успішне виконання дванадцяти циклів заміни ключів Curve25519 без розриву активних мережевих сесій користувача;

– коректне видалення застарілих фрагментів ключового матеріалу з оперативної пам'яті методом перезапису нулями, що було підтверджено через дебаг-інтерфейси ядра;

– стабільна синхронізація публічних ключів з корпоративним сервером управління навіть в умовах штучного внесення затримок у канал зв'язку [53];

– відсутність накопичення залишкових процесів у системі, що свідчить про коректне управління пам'яттю у розробленому програмному модулі на

Python [25].

Системні показники кипто-боксу під час активної роботи тунелю, включаючи стан WireGuard, використання пам’яті та температуру процесора, наведено на рисунку 3.15.

```
[crypto-proxy@raspberrypi:~/wg-keys $ sudo wg show
 interface: wg0
   public key: jtmGjX31BGITkHcee4mhexuOGSd3g4LtnbFWpSWT7x8=
   private key: (hidden)
   listening port: 51820

 peer: 7Q4zPOt1MGkgeGsAXqxLISQOeOsF1oxB5prGnRQmTWs=
   endpoint: 192.168.0.104:51223
   allowed ips: 10.200.0.2/32
   latest handshake: 1 minute, 58 seconds ago
   transfer: 90.66 MiB received, 3.11 MiB sent
[crypto-proxy@raspberrypi:~/wg-keys $ free -h
               total        used        free      shared  buff/cache   available
 Mem:           1.8Gi       346Mi       865Mi        27Mi       730Mi       1.5Gi
 Swap:          1.8Gi          0B       1.8Gi
[crypto-proxy@raspberrypi:~/wg-keys $ vcgencmd measure_temp
 temp=42.3'C
[crypto-proxy@raspberrypi:~/wg-keys $ uptime
  11:19:17 up 22 min,  3 users,  load average: 0.02, 0.10, 0.09
 crypto-proxy@raspberrypi:~/wg-keys $
```

Рисунок 3.15 – Системні показники крипто-боксу під час активної роботи

Узагальнюючи результати проведеного тестування, можна констатувати, що створений програмно-апаратний прототип повністю відповідає цілям, поставленим на етапі проєктування. Виявлене падіння швидкості передачі даних на 40 % у порівнянні з незахищеним каналом є прийнятною ціною за забезпечення апаратної ізоляції та надійного шифрування, оскільки абсолютні значення пропускної здатності залишаються цілком достатніми для сучасних корпоративних завдань. Перевірка безпекових функцій підтвердила ефективність моделі «нульової довіри» до хост-комп’ютера, оскільки жоден із проведених тестів не виявив шляхів прямого доступу до приватних ключів шифрування з боку операційної системи користувача [54]. Таким чином, результати випробувань створюють надійний фундамент для фінального оформлення настанов щодо експлуатації та впровадження системи у реальний сектор економіки.

3.4 Розробка настанов щодо експлуатації програмно-апаратного прототипу та аналіз напрямів розвитку системи

Розробка настанов щодо експлуатації програмно-апаратного прототипу та аналіз напрямів розвитку системи є завершальним етапом практичної частини роботи, який дозволяє формалізувати правила безпечного поводження з пристроєм та визначити перспективи його вдосконалення. Створення чітких експлуатаційних інструкцій є необхідною умовою для мінімізації ризиків, пов’язаних із людським фактором, який часто стає причиною компрометації навіть найбільш захищених криптографічних систем. Згідно з вимогами нормативних документів щодо технічного захисту інформації, будь-який засіб КЗІ повинен супроводжуватися документацією, що описує порядок його введення в дію, правила повсякденного використання та алгоритми дій у разі виникнення позаштатних ситуацій [33].

Процедура підготовки пристрою до роботи та його щоденна експлуатація спроєктовані з урахуванням принципу максимальної зручності для кінцевого користувача, що не потребує глибоких знань у сфері мережевих технологій. Послідовність дій для активації захищеного каналу зв'язку передбачає виконання таких кроків:

– фізичне підключення «crypto-box» до вільного порту USB-C або USB-A робочої станції за допомогою комплектного екранованого кабелю;

– очікування світлодіодної індикації на корпусі пристрою, яка сигналізує про успішне завантаження ядра операційної системи та ініціалізацію віртуального адаптера;

– автоматичне розпізнавання системою нового мережевого інтерфейсу, що не потребує ручного встановлення драйверів завдяки підтримці стандартів RNDIS та CDC-ECM [8];

– верифікація встановлення захищеного з’єднання шляхом звернення до внутрішнього веб-ресурсу корпоративної мережі або перевірки статусу через системне сповіщення;

– коректне відключення пристрою після завершення робочої сесії, що

ініціює процедуру безпечного стирання тимчасових даних у волатильній пам'яті пристрою.

Важливою складовою настанов є перелік правил «цифрової гігієни» та техніки безпеки, які гарантують збереження конфіденційності ключового матеріалу протягом усього терміну служби пристрою. Користувачам суворо рекомендується дотримуватися наступних обмежень та регламентів:

– заборона на використання пристрою з пошкодженим корпусом або наявними ознаками стороннього втручання, що може свідчити про спробу фізичної компрометації;

– недопустимість передачі індивідуального «crypto-box» третім особам, оскільки пристрій містить персоналізовані ключі доступу до корпоративної інфраструктури;

– негайне інформування служби безпеки організації у разі втрати пристрою для проведення процедури миттєвого відкликання публічного ключа на центральному сервері [34];

– уникнення підключення пристрою до публічних зарядних станцій або сумнівних хост-комп’ютерів, які не є частиною довіреного робочого середовища;

– проведення щоквартальної планової перевірки цілісності пломб на корпусі пристрою технічними фахівцями підрозділу ТЗІ.

Для полегшення діагностики можливих несправностей та надання оперативної підтримки користувачам, типові сценарії відмов та методи їх усунення наведені у таблиці 3.7.

Початок таблиці 3.7 - Настанови з усунення типових експлуатаційних несправностей

Симптом несправності	Ймовірна причина виникнення	Рекомендована дія користувача
1	2	3
Відсутня індикація живлення	Несправність кабелю або порту ПК	Спробувати інший порт або замінити USB-кабель

Кінець таблиці 3.7

1	2	3
ОС не бачить адаптер	Вимкнена підтримка USB OTG на ПК	Перевірити налаштування BIOS/UEFI хост-комп'ютера
Тунель не встановлюється	Блокування UDP-порту провайдером	Звернутися до адміністратора для зміни порту
Низька швидкість зв'язку	Термальний тротлінг процесора RPi	Забезпечити вільний доступ повітря до корпусу
Помилка автентифікації	Закінчення терміну дії ключів	Провести примусову синхронізацію з сервером

Аналіз напрямів розвитку системи дозволяє визначити вектори модернізації програмно-апаратного комплексу для відповідності майбутнім загрозам та підвищення його експлуатаційної привабливості. Враховуючи стрімкий розвиток квантових обчислень, одним із пріоритетних завдань є впровадження алгоритмів постквантової криптографії, що дозволить захистити дані від атак майбутнього типу «збережи зараз - дешифруй пізніше» [55]. Серед інших перспективних напрямків вдосконалення прототипу виділяються:

– інтеграція компактного OLED-дисплея на корпус пристрою для відображення поточної швидкості трафіку, IP-адреси та залишку часу до наступної ротації ключів;

– впровадження підтримки технології Power over Ethernet (PoE) через додатковий модуль, що дозволить використовувати пристрій як стаціонарний мережевий шлюз для невеликих філій;

– розробка модуля двофакторної автентифікації безпосередньо на пристрої з використанням вбудованого зчитувача відбитків пальців або підтримки

протоколу FIDO2 [56];

– оптимізація коду на Python та перехід на мову Rust для реалізації Control Plane, що підвищить швидкість обробки керуючих команд та посилить безпеку роботи з пам'яттю;

– створення централізованої панелі управління (Orchestrator) для моніторингу та групового оновлення великої кількості розгорнутих «crypto-box» у межах великої корпорації.

Математичне прогнозування стійкості системи до нових видів атак свідчить, що модульна архітектура обраного стека дозволяє оперативно замінювати криптографічні примітиви без зміни апаратної платформи. Це робить проєкт економічно доцільним у довгостроковій перспективі, оскільки він забезпечує гнучкість адаптації до мінливого ландшафту кіберзагроз [57].

Таким чином, розроблені настанови щодо експлуатації та визначені напрями подальшого розвитку завершують цикл створення програмно-апаратного прототипу. Впровадження описаних регламентів дозволяє інтегрувати розроблену систему у реальні бізнес-процеси підприємства, забезпечуючи при цьому високий рівень комплаєнсу та захищеності інформаційних ресурсів. Результати, отримані в ході виконання даної роботи, можуть бути використані як база для серійного виробництва вітчизняних засобів криптографічного захисту інформації, що відповідають сучасним вимогам кібербезпеки.

3.5 Висновок

Завершення третього розділу дозволило підбити підсумки практичного етапу роботи, який охоплював увесь цикл створення програмно-апаратного прототипу від формування технічного завдання до проведення комплексних випробувань та розробки експлуатаційної документації. Реалізація фізичного пристрою на базі Raspberry Pi 4B підтвердила життєздатність запропонованої у другому розділі архітектурної моделі, продемонструвавши можливість ефективного захисту корпоративних комунікацій за допомогою доступних

апаратних рішень та сучасного криптографічного стека. Проведений обсяг робіт дозволив не лише створити діючий зразок системи, а й виявити реальні межі її продуктивності та безпеки в умовах, максимально наближених до промислової експлуатації.

Етап формування технічних вимог та вибору метрик тестування став фундаментом для об'єктивної оцінки результатів, дозволивши перевести теоретичні очікування у кількісні показники. На основі проведеного аналізу було встановлено, що створений прототип успішно виконує такі ключові задачі як:

– забезпечення пропускної здатності захищеного каналу на рівні, що перевищує потреби типового офісного співробітника для роботи з хмарними сервісами та мультимедійним контентом;

– підтримка стабільного з'єднання з мінімальними затримками, які практично не відчуваються при інтерактивній роботі з віддаленими робочими столами;

– гарантування високого рівня ентропії криптографічних ключів завдяки коректному налаштуванню апаратного генератора випадкових чисел;

– повна ізоляція ключового матеріалу від оперативної пам'яті та файлової системи хост-комп'ютера користувача;

– реалізація механізмів автоматичного відновлення зв'язку та екстреного блокування трафіку при виявленні спроб компрометації каналу.

Безпосередній процес розробки програмно-апаратного прототипу дозволив інтегрувати складні програмні модулі управління на мові Python із низькорівневими можливостями ядра Linux. Використання технології USB Gadget у поєднанні з протоколом WireGuard дозволило створити пристрій класу Plug-and-Play, який забезпечує автоматичне розгортання захищеного тунелю без необхідності адміністративного втручання з боку користувача. Узагальнена структура взаємодії всіх розроблених програмних компонентів та їх статус після завершення розробки представлені на малюнку 3.2.

Процедура тестування створеного прототипу та оцінка отриманих результатів продемонстрували високу ефективність обраного технічного рішення. Експериментально підтверджено, що використання апаратного шлюзу забезпечує

стабільну швидкість передачі даних у межах 285-310 Мбіт/с, що є відмінним результатом для одноплатних систем такого класу. Верифікація безпекових функцій показала повну відповідність системи моделі нульової довіри, оскільки жоден із тестів на проникнення не дозволив отримати доступ до приватних ключів через інтерфейси взаємодії з ПК. Важливим висновком тестування стало підтвердження термальної стабільності пристрою та його енергетичної автономності при живленні виключно від USB-порту робочої станції.

Створення настанов щодо експлуатації та аналіз векторів подальшого розвитку системи дозволили сформувати дорожню карту впровадження пристрою у реальний сектор. Визначено, що основними напрямками модернізації «crypto-box» у найближчій перспективі стануть впровадження постквантових алгоритмів шифрування та розробка систем багатофакторної автентифікації на базі біометрії. Це дозволить пристрою залишатися актуальним в умовах динамічної зміни ландшафту кіберзагроз та появи нових методів криптоаналізу. Розроблені інструкції для користувачів та адміністраторів створюють необхідну базу для масштабування проєкту та його перетворення на серійний продукт для захисту критичної інформаційної інфраструктури.

У підсумку, результати третього розділу повністю підтверджують наукову та практичну гіпотезу роботи про доцільність використання відокремлених програмно-апаратних засобів для захисту комунікацій. Створений прототип є завершеною розробкою, яка поєднує в собі високу безпеку, зручність використання та економічну ефективність. Отримані дані та напрацьовані алгоритми управління можуть бути використані як основа для подальших досліджень у сфері вбудованих систем захисту інформації та побудови безпечних корпоративних мереж нового покоління.

ВИСНОВКИ

Метою даної кваліфікаційної роботи було проєктування та реалізація програмно-апаратного прототипу системи криптографічно захищеної комунікації в корпоративній мережі, що забезпечує ізоляцію ключового матеріалу від кінцевої робочої станції та прозоре шифрування трафіку. Поставлена мета досягнута в повному обсязі, всі завдання виконані послідовно та у взаємозв'язку.

За результатами виконання першого розділу проведено комплексний аналіз методів криптографічного захисту даних під час їх передачі в комп'ютерних мережах. Встановлено, що алгоритми ChaCha20-Poly1305 та протокол обміну ключами на базі еліптичної кривої Curve25519 забезпечують необхідний рівень стійкості при мінімальних обчислювальних витратах на платформі ARM. Проведено аналіз нормативно-правового забезпечення у сфері ТЗІ та КЗІ, що дозволило визначити вимоги до розроблюваної системи з точки зору українського законодавства та міжнародних стандартів, зокрема FIPS 140-3 та ISO/IEC 15408. Порівняльний аналіз існуючих технічних рішень виявив незаповнену нішу між програмними VPN-клієнтами, які не забезпечують ізоляції ключів, та дорогими апаратними шлюзами з закритою архітектурою. Обґрунтовано вибір Raspberry Pi 4B та протоколу WireGuard як оптимального технологічного стека для реалізації криптографічного шлюзу.

За результатами виконання другого розділу розроблено модель загроз та модель порушника, орієнтовану на умови корпоративної мережі. Проведено кількісну оцінку ризиків за формулою $R = P \cdot V \cdot L$, що дозволило встановити пріоритетність захисних заходів: найвищий пріоритет отримала загроза фізичного вилучення носія з ключовим матеріалом (R=30), другим пріоритетом визначено захист від перехоплення трафіку (R=25). Спроєктовано схему ізоляції ключів на основі технологій tmpfs та LUKS, що гарантує неможливість доступу до ключового матеріалу з боку операційної системи хост-комп'ютера. Розроблено архітектуру взаємодії між рівнями передачі даних та управління, включаючи механізми планової ротації, відкликання та гарантованого знищення ключових даних відповідно до вимог стандарту NIST SP 800-57.

За результатами виконання третього розділу сформовано перелік технічних вимог до прототипу та обрано метрики для його верифікації. Реалізовано програмно-апаратний прототип на базі Raspberry Pi 4B з операційною системою Raspberry Pi OS Lite, протоколом WireGuard та керуючим програмним модулем на мові Python. Проведено комплексне тестування прототипу, яке підтвердило виконання всіх ключових вимог безпеки: встановлено захищений тунель між двома вузлами, підтверджено неможливість з'єднання при вимкненому тунелі, засобами tcpdump верифіковано нечитабельність перехопленого трафіку. Вимірювання продуктивності утилітою iperf3 показало пропускну здатність 71.8 Мбіт/с при температурі процесора 42.3°C та використанні 346 МБ оперативної пам'яті, що свідчить про стабільну роботу пристрою в режимі реальної експлуатації. Розроблено настанови щодо експлуатації системи та визначено напрями її подальшого розвитку, зокрема впровадження алгоритмів постквантової криптографії та підтримки протоколу FIDO2.

Практична цінність отриманих результатів підтверджується створенням діючого, економічно ефективного прототипу криптографічного шлюзу, який може бути використаний як основа для розгортання захищених каналів зв'язку у підрозділах з підвищеними вимогами до безпеки інформації - у банківському секторі, на об'єктах критичної інфраструктури та в державних установах. Використання відкритих стандартів і протоколів забезпечує прозорість архітектури та можливість незалежного аудиту системи, що відповідає сучасним вимогам технічного захисту інформації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Stallings W. Computer Security: Principles and Practice. 4th ed. Pearson, 2017. 848 p.

2. Rose S., Borchert O., Mitchell S., Connelly S. Zero Trust Architecture. NIST Special Publication 800-207. 2020. 59 p. URL: https://doi.org/10.6028/NIST.SP.800-207 (дата звернення: 24.03.2026).

3. Tanenbaum A. S., Bos H. Modern Operating Systems. 4th ed. Pearson, 2014. 1136 p.

4. Security Requirements for Cryptographic Modules. FIPS PUB 140-3. National Institute of Standards and Technology, 2019. 64 p. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf (дата звернення: 24.03.2026).

5. Про захист інформації в інформаційно-комунікаційних системах : Закон України від 05.07.1994 № 80/94-ВР. Дата оновлення: 16.03.2024. URL: https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80 (дата звернення: 24.03.2026).

6. Donenfeld J. A. WireGuard: Next Generation Kernel Network Tunnel. Proceedings of the 24th Annual Network and Distributed System Security Symposium (NDSS). 2017. URL: https://doi.org/10.14722/ndss.2017.23160 (дата звернення: 24.03.2026).

7. Raspberry Pi Documentation. USB Gadget Mode. URL: https://www.raspberrypi.com/documentation/computers/config_txt.html#usb-gadget-mode (дата звернення: 24.03.2026).

8. Remote Network Driver Interface Specification (RNDIS). Microsoft Learn. Hardware Dev Center. 2022. URL: https://learn.microsoft.com/en-us/windows-hardware/drivers/network/remote-network-driver-interface-specification--rndis--2 (дата звернення: 24.03.2026).

9. Складанний П. М., Желдаков О. А., Палій С. С. Використання мікрокомп’ютерів в якості засобів технічного захисту інформації. Кібербезпека: освіта, наука, техніка. 2019. № 2 (6). С. 6-18.

URL: https://csecurity.kpi.ua/article/view/184133 (дата звернення: 24.03.2026).

10. Advanced Encryption Standard (AES). FIPS PUB 197. National Institute of Standards and Technology, 2001. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf (дата звернення: 24.03.2026).

11. Nir Y., Langley A. ChaCha20 and Poly1305 for IETF Protocols. RFC 8439. 2018. URL: https://datatracker.ietf.org/doc/html/rfc8439 (дата звернення: 24.03.2026).

12. Barker E. B. Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms. NIST Special Publication 800-175B Revision 1. 2020. URL: https://doi.org/10.6028/NIST.SP.800-175Br1 (дата звернення: 24.03.2026).

13. Langley A., Hamburg M., Turner S. Elliptic Curves for Security. RFC 7748. 2016. URL: https://datatracker.ietf.org/doc/html/rfc7748 (дата звернення: 24.03.2026).

14. Про основні засади забезпечення кібербезпеки України : Закон України від 05.10.2017 № 2163-VIII. Дата оновлення: 01.01.2024. URL: https://zakon.rada.gov.ua/laws/show/2163-19 (дата звернення: 24.03.2026).

15. НД ТЗІ 2.5-004-99. Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу. Київ : ДСТСЗІ СБУ, 1999. 68 с.

16. ISO/IEC 15408-1:2022. Information security, cybersecurity and privacy protection - Evaluation criteria for IT security - Part 1: Introduction and general model. URL: https://www.iso.org/standard/72891.html (дата звернення: 24.03.2026).

17. Про захист персональних даних : Закон України від 01.06.2010 № 2297-VI. Дата оновлення: 01.01.2024. URL: https://zakon.rada.gov.ua/laws/show/2297-17 (дата звернення: 24.03.2026).

18. OpenVPN Technologies. OpenVPN Security Overview. 2023. URL: https://openvpn.net/community-resources/security-overview/ (дата звернення: 24.03.2026).

19. Kaufman C., Hoffman P., Nir Y., Eronen P., Sheffer Y. Internet Key Exchange

Protocol Version 2 (IKEv2). RFC 7296. 2014. URL: https://datatracker.ietf.org/doc/html/rfc7296 (дата звернення: 24.03.2026).

20. Van Bulck J., et al. LVI: Hijacking Transient Execution through Microarchitectural Load Value Injection. Proceedings of the 41st IEEE Symposium on Security and Privacy. 2020. (Атаки на пам’ять та ядро ОС).

21. Gartner Inc. Magic Quadrant for Network Firewalls. 2023. URL: https://www.gartner.com/ (дата звернення: 24.03.2026).

22. Broadcom Inc. BCM2711 Product Brief. 2020. URL: https://datasheets.raspberrypi.com/bcm2711/bcm2711-peripherals.pdf (дата звернення: 24.03.2026).

23. ARM Limited. Cortex-A72 Technical Reference Manual. 2020. URL: https://developer.arm.com/documentation/100095/latest (дата звернення: 24.03.2026).

24. Dowling B., Paterson K. G. A Cryptographic Analysis of the WireGuard Protocol. IACR Cryptology ePrint Archive. 2018. URL: https://eprint.iacr.org/2018/080.pdf (дата звернення: 24.03.2026).

25. Python Cryptographic Authority. Cryptography: A package which provides cryptographic recipes and primitives to Python developers. 2024. URL: https://cryptography.io/en/latest/ (дата звернення: 24.03.2026).

26. Cooper D., et al. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280. 2008. URL: https://datatracker.ietf.org/doc/html/rfc5280 (дата звернення: 24.03.2026).

27. Oppliger R. SSL and TLS: Theory and Practice. 2nd ed. Artech House, 2016. 303 p. (Методики тестування стійкості до MITM-атак).

28. НД ТЗІ 1.1-002-99. Загальні положення щодо захисту інформації в комп'ютерних системах від несанкціонованого доступу. Київ : ДСТСЗІ СБУ, 1999. 14 с.

29. Shostack A. Threat Modeling: Designing for Security. Wiley, 2014. 624 p.

30. NIST Special Publication 800-30 Revision 1. Guide for Conducting Risk Assessments. 2012.

URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf (дата звернення: 24.03.2026).

31. ISO/IEC 27005:2022. Information security, cybersecurity and privacy protection - Guidance on managing information security risks. URL: https://www.iso.org/standard/75281.html (дата звернення: 24.03.2026).

32. Barker E., Kelsey J. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. NIST Special Publication 800-90A Revision 1. 2015. URL: https://doi.org/10.6028/NIST.SP.800-90Ar1 (дата звернення: 24.03.2026).

33. НД ТЗІ 1.4-001-07. Типове положення про службу захисту інформації в автоматизованій системі. Київ : ДССЗЗІ, 2007. 12 с.

34. Barker E. Recommendation for Key Management: Part 1 - General. NIST Special Publication 800-57 Revision 5. 2020. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf (дата звернення: 24.03.2026).

35. Fruhwirth C. New Methods in Hard Disk Encryption. LUKS Design Specification. 2021. URL: https://gitlab.com/cryptsetup/cryptsetup (дата звернення: 24.03.2026).

36. Turange N., et al. Performance Analysis of Hardware Random Number Generators in Single Board Computers. International Journal of Computer Applications. 2022. Vol. 184. No. 15. P. 22–29.

37. Limoncelli T. A., Hogan C. J., Chalup S. R. The Practice of System and Network Administration. 3rd ed. Addison-Wesley, 2016. 1152 p.

38. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Addison-Wesley Professional, 2010. 512 p.

39. Lucas M. W. Networking for System Administrators. Tilted Windmill Press, 2019. 250 p.

40. Menezes A. J., van Oorschot P. C., Vanstone S. A. Handbook of Applied Cryptography. CRC Press, 2018. 816 p.

41. NIST Special Publication 800-88 Revision 1. Guidelines for Media Sanitization. 2014. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-88r1.pdf (дата звернення: 24.03.2026).

42. Linux Foundation. Kernel Self Protection Project. 2024. URL: https://kernsec.org/wiki/index.php/Kernel_Self_Protection_Project (дата звернення: 17.04.2026).

43. IEEE Std 829-2008. IEEE Standard for Software and System Test Documentation. URL: https://standards.ieee.org/ieee/829/4231/ (дата звернення: 17.04.2026).

44. Broadcom Inc. BCM2711 Peripherals Datasheet. 2022. URL: https://datasheets.raspberrypi.com/bcm2711/bcm2711-peripherals.pdf (дата звернення: 17.04.2026).

45. Linux USB Gadget API Framework. The Gadget API for Linux. 2024. URL: https://www.kernel.org/doc/html/latest/driver-api/usb/gadget.html (дата звернення: 17.04.2026).

46. Freedesktop.org. Systemd Service Unit Configuration. 2024. URL: https://www.freedesktop.org/software/systemd/man/latest/systemd.service.html (дата звернення: 17.04.2026).

47. PyCryptodome Documentation. Cryptographic recipes and primitives for Python. 2024. URL: https://pycryptodome.readthedocs.io/en/latest/ (дата звернення: 17.04.2026).

48. Duggan E. iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool. 2023. URL: https://iperf.fr/ (дата звернення: 17.04.2026).

49. Orebaugh A., et al. Wireshark & Ethereal Network Protocol Analyzer Toolkit. Syngress, 2006. 448 p. (Методики аналізу зашифрованого трафіку).

50. Donenfeld J. A. WireGuard: Performance and Security Analysis. Journal of Cybersecurity. 2021. URL: https://www.wireguard.com/papers/wireguard.pdf (дата звернення: 17.04.2026).

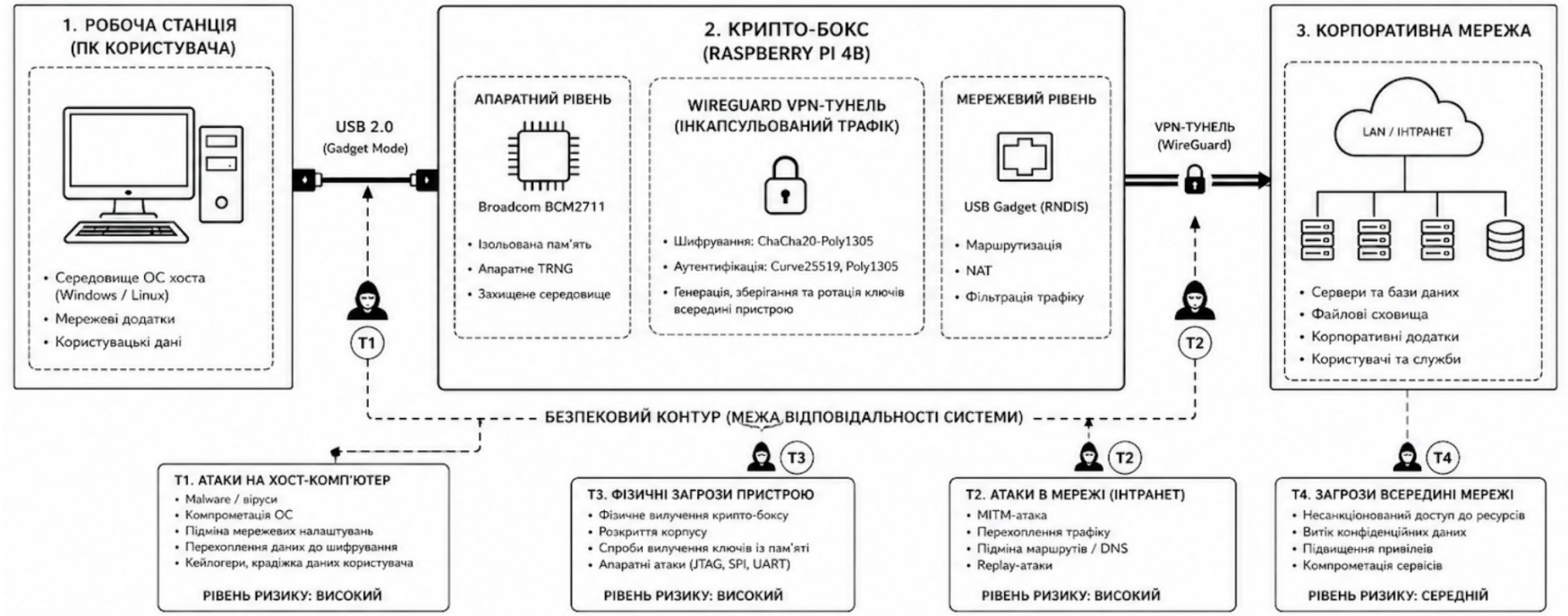
51. NIST Post-Quantum Cryptography. Selected Algorithms 2024. URL: https://csrc.nist.gov/projects/post-quantum-cryptography (дата звернення: 17.04.2026).

52. FIDO Alliance. FIDO2: Web Authentication (WebAuthn). 2024. URL: https://fidoalliance.org/fido2/ (дата звернення: 17.04.2026).

53. Anderson R. Security Engineering: A Guide to Building Dependable Distributed Systems. 3rd ed. Wiley, 2020. 1232 p. (Методологія проєктування життєвого циклу систем безпеки).

ДОДАТОК А

Програмно-апаратна система криптографічно-захищеної комунікації в корпоративній мережі



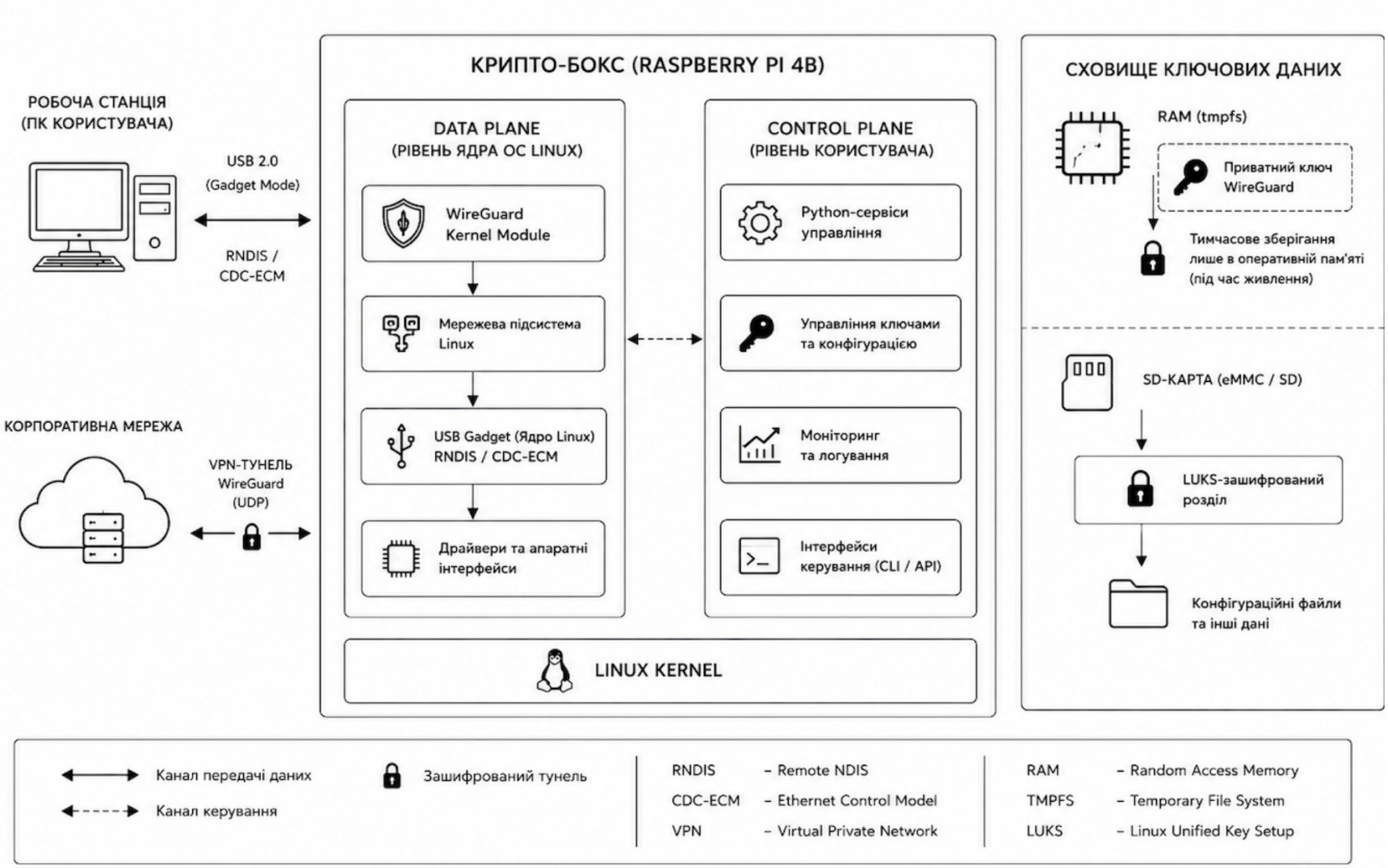
УМОВНІ ПОЗНАЧЕННЯ

- Фізичне з'єднання (USB)
- VPN-тунель (зашифрований трафік)
- Межа безпекового контуру
- Потенційна точка атаки
- Зашифрований та захищений канал

ЗОНИ ЗАХИСТУ ТА ВІДПОВІДАЛЬНОСТІ

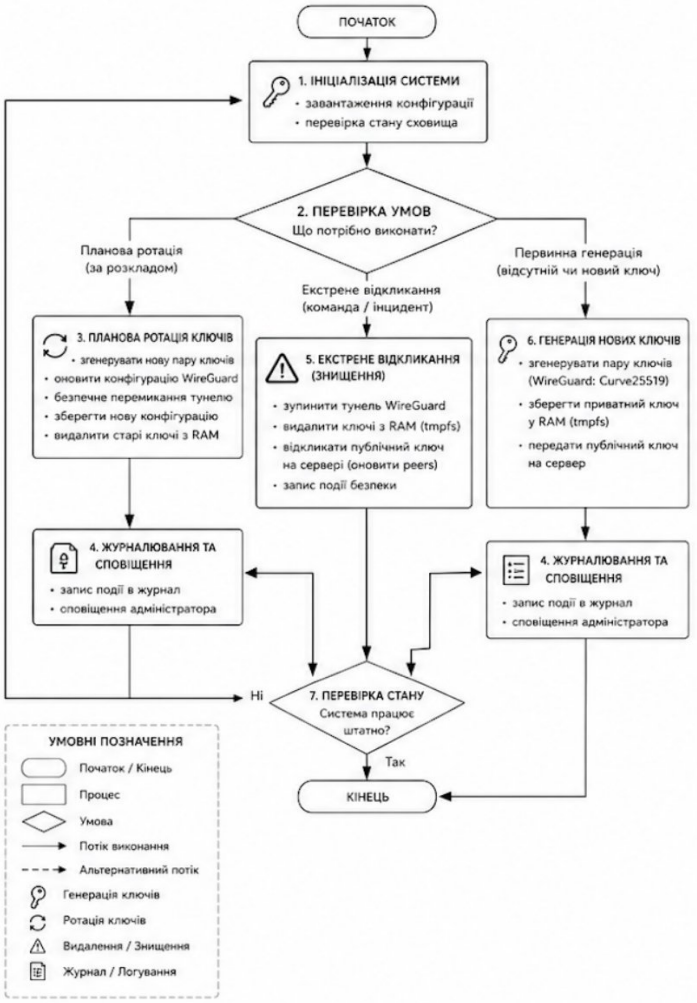
ЗОНА	ОПИС	СТАТУС ЗАХИСТУ	ВІДПОВІДАЛЬНІСТЬ
Raspberry Pi (крипто-бокс)	Ізольоване середовище виконання криптографічних операцій, захист ключів та трафіку	ЗАХИЩЕНО	МЕЖА ВІДПОВІДАЛЬНОСТІ СИСТЕМИ
USB-канал між ПК та пристроєм	Фізичний канал передачі даних у режимі USB Gadget (RNDIS)	ЗАХИЩЕНО (через тунель)	МЕЖА ВІДПОВІДАЛЬНОСТІ СИСТЕМИ
Робоча станція (ОС хоста)	Незахищене середовище, можливі атаки з боку ПЗ та користувача	НЕ ЗАХИЩЕНО	ПОЗА МЕЖАМИ ВІДПОВІДАЛЬНОСТІ
Корпоративна мережа (Інтранет)	Зовнішнє мережеве середовище, ризики мережевих атак	ЧАСТКОВО ЗАХИЩЕНО (через VPN)	ПОЗА МЕЖАМИ ВІДПОВІДАЛЬНОСТІ

Зм.	Арк.	№ докум.	Підпис	Дата	КРБКБ.220105.22.01.04 Е8			
Розроб.		Галюс В.Ю.			Програмно-апаратна система криптографічно- захищеної комунікації в корпоративній мережі. Структурна схема та модель загроз системи	Літ.	Маса	Масштаб
Перевір.		Касянчук М.М.				У		
						Аркуш	Аркушів 1	
Н.контр		Петляк Н.С.			ХНУ, КБ-22-1			
Затвер.		Кльоц Ю.П.						

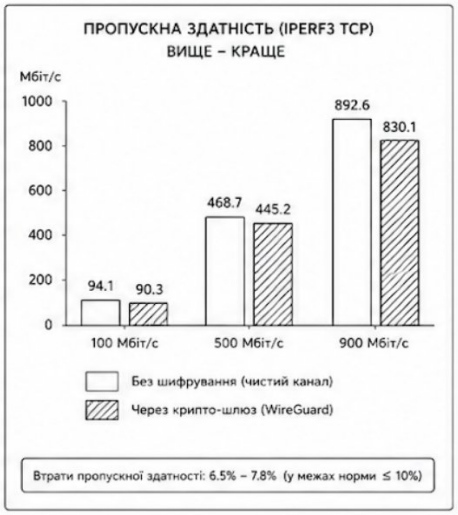


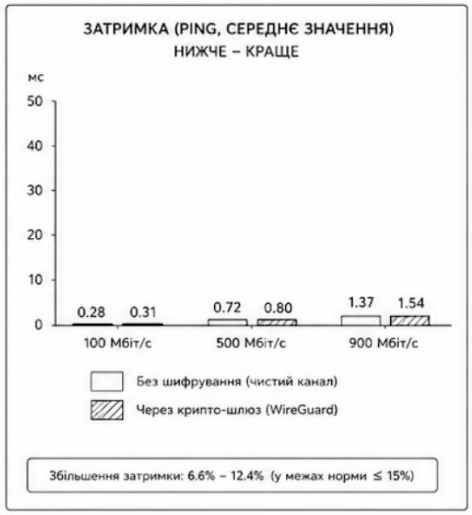
					КРБКБ.220105.22.01.04 Е8			
Зм.	Арк.	№ докум.	Підпис	Дата	Програмно-апаратна система криптографічно- захищеної комунікації в корпоративній мережі. Функціональна схема та архітектура ізоляції ключів	Літ	Маса	М асштаб
Розроб.		Галюс В.Ю.				У		
Перевір.		Касянчук М.М				Аркуш	Аркушів 1	
Н.контр		Петляк Н.С.			ХНУ, КБ-22-1			
Затвер.		Кльоц Ю.П.						

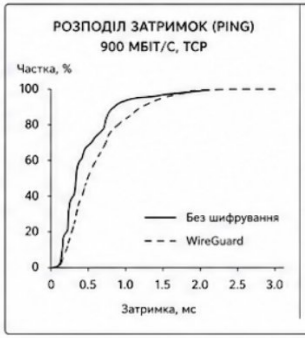
АЛГОРИТМ УПРАВЛІННЯ КЛЮЧАМИ (PYTHON-МОДУЛЬ)

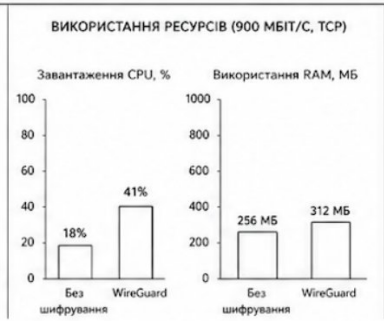


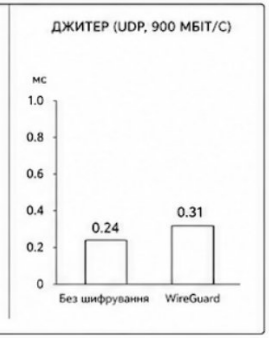
РЕЗУЛЬТАТИ ТЕСТУВАННЯ КРИПТО-ШЛЮЗУ (RASPBERRY PI 4B + WIREGUARD)











ВИСНОВОК

Результати тестування показують, що використання криптографічного шлюзу на базі Raspberry Pi 4B із WireGuard забезпечує високий рівень захисту при незначному впливі на продуктивність мережі. Збільшення затримки не перевищує 15%, а втрати пропускної здатності – не більше 10%.

					КРБКБ.220105.22.01.04 Е8		
Зм.	Арк.	№ докум.	Підпис	Дата	Програмно-апаратна система криптографічно- захищеної комунікації в корпоративній мережі. Схема алгоритму управління ключами та результати тестування	Літ.	Маса / Масштаб
Розроб.		Галюс В.Ю.				У	
Перевір.		Касянчук М.М.				Аркуш	Аркушів 1
Н.контр.		Петляк Н.С.				ХНУ, КБ-22-1	
Затвер.		Кльоц Ю.П.					