МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Кваліфікаційна наукова праця
на правах рукопису

Науменко Сергій Васильович

ДИСЕРТАЦІЯ

МЕТОД ТА МОДЕЛІ ЗАХИСТУ ІНФОРМАЦІЇ ДЛЯ КІБЕРФІЗИЧНИХ СИСТЕМ З ОБМЕЖЕНИМИ РЕСУРСАМИ

123 – Комп’ютерна інженерія
Подається на здобуття ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
С.В. НАУМЕНКО

Науковий керівник:
Розломій Інна Олександрівна
кандидат технічних наук, доцент

Черкаси – 2026

АНОТАЦІЯ

Науменко С.В. Метод та моделі захисту інформації для кіберфізичних систем з обмеженими ресурсами. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 123 Комп'ютерна інженерія. – Черкаський державний технологічний університет, Черкаси, 2026.

Дисертаційна робота присвячена розв'язанню актуальної науково-технічної проблеми підвищення рівня криптографічного захисту інформації в кіберфізичних системах (КФС), функціонування яких відбувається в умовах обмежених обчислювальних ресурсів, енергоспоживання та жорстких часових обмежень. Актуальність дослідження зумовлена стрімким поширенням кіберфізичних систем у промисловості, медицині, транспорті, енергетиці та інших критично важливих галузях, а також зростанням кількості кібератак на сенсорні вузли, вбудовані контролери, периферійні обчислювальні модулі та канали взаємодії між компонентами таких систем.

Особливістю кіберфізичних систем є тісна інтеграція обчислювальних, мережевих і фізичних компонентів, що зумовлює необхідність комплексного підходу до криптографічного захисту інформації на всіх рівнях їх функціонування. У той же час, більшість компонентів КФС мають суттєві обмеження за обсягом пам'яті, продуктивністю процесора та енергоспоживанням, що ускладнює використання класичних криптографічних алгоритмів і протоколів без втрати працездатності системи або зростання затримок обробки даних.

У роботі проведено аналіз сучасного стану предметної області криптографічного захисту інформації в кіберфізичних системах, виконано систематизацію архітектурних рівнів КФС відповідно до семирівневої моделі та здійснено класифікацію загроз інформаційній безпеці з урахуванням

особливостей сенсорних модулів, вбудованих пристроїв, мережевої взаємодії, edge/fog-обчислень і хмарних сервісів. Показано, що існуючі підходи до криптографічного захисту не завжди враховують специфіку ресурсних обмежень, динамічність топології та вимоги до реального часу, що зумовлює необхідність розробки адаптованих методів і моделей захисту.

Обґрунтовано доцільність застосування полегшених криптографічних механізмів у компонентах кіберфізичних систем та сформульовано вимоги до їх вибору з урахуванням продуктивності, енергоспоживання, використання пам’яті, криптографічної стійкості та відповідності сучасним стандартам інформаційної безпеки. Розглянуто сучасні полегшені симетричні алгоритми шифрування, хеш-функції та механізми автентифікованого шифрування, а також проаналізовано обмеження застосування класичних алгоритмів асиметричної криптографії в ресурсообмежених середовищах.

У дисертації розроблено модель криптографічного захисту інформації на рівні сенсорних вузлів і вбудованих пристроїв, яка включає механізми перевірки цілісності та автентичності програмного коду, захищеного завантаження (Secure Boot), управління ключовим матеріалом і протидії атакам на початкових етапах запуску системи. Запропоновано підхід до структуризації та реалізації криптографічних процедур з урахуванням обмежень архітектури мікроконтролерів та особливостей операційних систем реального часу.

Побудовано моделі захищеного інформаційного обміну між компонентами кіберфізичних систем у мережевих та edge-архітектурах, орієнтовані на забезпечення конфіденційності, цілісності та автентичності даних за умов нестабільних каналів зв’язку та обмежених ресурсів обчислення. Запропоновано метод формування критеріїв і метрик оцінювання ефективності криптографічного захисту, що охоплюють часові характеристики виконання алгоритмів, рівень завантаження процесора, споживання енергії, використання

оперативної та постійної пам’яті, стабільність часу виконання та стійкість до побічних атак.

На основі розроблених моделей запропоновано метод побудови комплексної системи криптографічного захисту інформації в кіберфізичних системах, який інтегрує окремі механізми захисту в єдину архітектуру з урахуванням взаємодії фізичних, мережевих і сервісних рівнів. Метод передбачає адаптивний вибір криптографічних алгоритмів залежно від ресурсних можливостей компонентів КФС та умов їх функціонування.

Запропонований метод реалізовано у програмно-апаратному середовищі з використанням мікроконтролера STM32. Проведено експериментальні дослідження продуктивності, енергоспоживання та використання пам’яті для сучасних полегшених криптографічних механізмів при реалізації в середовищах FreeRTOS та bare-metal. Отримані результати дозволили виконати порівняльний аналіз з традиційними криптографічними підходами та підтвердили ефективність запропонованих рішень в умовах обмежених ресурсів. Експериментальні дослідження показали, що:

- перевірка цілісності та автентичності програмного коду виконується за час до 120 мс для прошивки обсягом до 128 КБ при використанні не більше 5 КБ оперативної пам’яті;
- застосування адаптивного вибору криптографічних алгоритмів дозволяє зменшити енергоспоживання криптографічних операцій у середньому на 23% порівняно з використанням AES-128;
- організація захищеного обміну даними в мережі з 20 вузлів забезпечує затримку передачі до 1,4 мс при збереженні стабільності з’єднання в умовах зміни маршрутизації;
- використання запропонованих підходів дозволяє знизити обчислювальне навантаження на 35–40% та підвищити продуктивність системи на 20–25% у порівнянні з традиційними рішеннями.

Наукова новизна дисертаційної роботи полягає у подальшому розвитку методів криптографічного захисту інформації в кіберфізичних системах шляхом розробки моделей і метрик оцінювання ефективності полегшених криптографічних механізмів з урахуванням обчислювальних та енергетичних обмежень, а також у створенні узагальненого методу побудови комплексної системи захисту, орієнтованої на багаторівневу архітектуру КФС. Теоретичні результати обґрунтовано та підтверджено експериментальними дослідженнями.

Практичне значення одержаних результатів полягає у можливості їх використання при проєктуванні та впровадженні систем криптографічного захисту в кіберфізичних системах, зокрема в IoT-пристроях, медичних вбудованих системах, промислових контролерах та периферійних обчислювальних платформах.

Основні результати дисертаційної роботи опубліковано у наукових працях у фахових виданнях України та виданнях, індексованих у міжнародних наукометричних базах, а також апробовано на міжнародних і всеукраїнських науково-практичних конференціях.

Ключові слова: кіберфізичні системи, Інтернет речей, полегшена криптографія, безпека даних, вбудовані пристрої, сенсорні вузли, Secure Boot, шифрування, edge-обчислення, хмарні обчислення, автентифікація, хешування.

SUMMARY

Naumenko S.V. Information protection method and models for cyber-physical systems with limited resources. – Qualifying scientific work on the rights of the manuscript.

Dissertation for the degree of Doctor of Philosophy in specialty 123 – Computer Engineering. – Cherkasy State Technological University, Cherkasy, 2026.

The dissertation work is devoted to solving the urgent scientific and technical problem of increasing the level of cryptographic information protection in cyber-physical systems (CPS), the functioning of which occurs under conditions of limited computing resources, energy consumption and strict time constraints. The relevance of the research is due to the rapid spread of cyber-physical systems in industry, medicine, transport, energy and other critical industries, as well as the increase in the number of cyberattacks on sensor nodes, embedded controllers, peripheral computing modules and interaction channels between components of such systems.

A feature of cyber-physical systems is the close integration of computing, network and physical components, which necessitates the need for a comprehensive approach to cryptographic information protection at all levels of their functioning. At the same time, most components of the CPS have significant limitations in terms of memory capacity, processor performance and power consumption, which makes it difficult to use classical cryptographic algorithms and protocols without losing system performance or increasing data processing delays.

The paper analyzes the current state of the subject area of cryptographic information protection in cyber-physical systems, systematizes the architectural levels of the CPS according to the seven-level model, and classifies threats to information security taking into account the features of sensor modules, embedded devices, network interaction, edge/fog computing and cloud services. It is shown that existing approaches to cryptographic protection do not always take into account the specifics of resource

constraints, topology dynamics and real-time requirements, which necessitates the development of adapted methods and protection models.

The feasibility of using lightweight cryptographic mechanisms in components of cyber-physical systems is substantiated and the requirements for their selection are formulated taking into account performance, power consumption, memory usage, cryptographic stability and compliance with modern information security standards. Modern lightweight symmetric encryption algorithms, hash functions and authenticated encryption mechanisms are considered, and the limitations of the use of classical asymmetric cryptography algorithms in resource-limited environments are analyzed.

The dissertation develops a model of cryptographic information protection at the level of sensor nodes and embedded devices, which includes mechanisms for verifying the integrity and authenticity of the program code, secure boot, key material management and countering attacks at the initial stages of system startup. An approach to structuring and implementing cryptographic procedures is proposed, taking into account the limitations of the microcontroller architecture and the features of real-time operating systems.

Models of secure information exchange between components of cyber-physical systems in network and edge architectures are built, focused on ensuring confidentiality, integrity and authenticity of data under conditions of unstable communication channels and limited computing resources. A method for forming criteria and metrics for evaluating the effectiveness of cryptographic protection is proposed, covering the time characteristics of algorithm execution, processor load level, energy consumption, use of RAM and non-RAM, stability of execution time and resistance to side attacks.

Based on the developed models, a method for building a comprehensive system of cryptographic information protection in cyber-physical systems is proposed, which integrates individual protection mechanisms into a single architecture taking into account the interaction of physical, network and service levels. The method involves

adaptive selection of cryptographic algorithms depending on the resource capabilities of the CPS components and their operating conditions.

The proposed method is implemented in a software-hardware environment using the STM32 microcontroller. Experimental studies of performance, power consumption and memory usage for modern lightweight cryptographic mechanisms when implemented in FreeRTOS and bare-metal environments were conducted. The results obtained allowed for a comparative analysis with traditional cryptographic approaches and confirmed the effectiveness of the proposed solutions in resource-constrained conditions. Experimental studies showed that:

- verification of the integrity and authenticity of the program code is performed in up to 120 ms for firmware up to 128 KB using no more than 5 KB of RAM;
- application of adaptive selection of cryptographic algorithms allows reducing the power consumption of cryptographic operations by an average of 23% compared to using AES-128;
- organization of secure data exchange in a network of 20 nodes provides a transmission delay of up to 1.4 ms while maintaining connection stability in conditions of routing changes;
- using the proposed approaches allows reducing the computational load by 35–40% and increasing system performance by 20–25% compared to traditional solutions.

The scientific novelty of the dissertation work lies in the further development of methods for cryptographic information protection in cyber-physical systems by developing models and metrics for assessing the effectiveness of lightweight cryptographic mechanisms taking into account computational and energy constraints, as well as in creating a generalized method for building a comprehensive protection system focused on the multi-level architecture of the CPS. The theoretical results are substantiated and confirmed by experimental studies.

The practical significance of the results obtained lies in the possibility of their use in the design and implementation of cryptographic protection systems in cyber-physical systems, in particular in IoT devices, medical embedded systems, industrial controllers and peripheral computing platforms.

The main results of the dissertation work have been published in scientific papers in professional publications of Ukraine and publications indexed in international scientometric databases, and have also been tested at international and all-Ukrainian scientific and practical conferences.

Keywords: cyber-physical systems, Internet of Things, lightweight cryptography, data security, embedded devices, sensor nodes, Secure Boot, encryption, edge computing, cloud computing, authentication, hashing.

Список публікацій здобувача

[1] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. IoT Smart Implants: Information Security and the Implementation of Lightweight Cryptography. *Proceedings of the 6th International Conference on Informatics & Data-Driven Medicine (IDDM'2023).* 2023. P. 145–146. URL: https://ceur-ws.org/Vol-3609/paper12.pdf **(Scopus)**

[2] Rozlomii I., Yarmilko A., Naumenko S. Data security of IoT devices with limited resources: challenges and potential solutions. *Proceedings of the 4th Edge Computing Workshop (DOORS 2024).* 2024. Vol. 3666. P. 85–96. URL: https://ceur-ws.org/Vol-3666/paper13.pdf **(Scopus)**

[3] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The role of encryption in information protection for cloud computing. *IEEE 4th International Conference on Smart Information Systems and Technologies (SIST).* 2024. P. 70–75. URL: https://doi.org/10.1109/SIST61555.2024.10629501 **(Scopus)**

[4] Yarmilko A., Rozlomii I., Naumenko S. Dependability of embedded systems in the Industrial Internet of Things: Information security and reliability of the

communication cluster. *Information Technology for Education, Science, and Technics.* 2024. Vol. 222. P. 235–249. URL: https://doi.org/10.1007/978-3-031-71804-5_16 **(Scopus)**

[5] Rozlomii I., Naumenko S., Mykhailovskyi P., Monarkh V. Resource-saving cryptography for microcontrollers in biomedical devices. *IEEE 5th KhPI Week on Advanced Technology (KhPIWeek).* 2024. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61434.2024.10877958 **(Scopus)**

[6] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. Hardware encryptors and cryptographic libraries for optimizing security in IoT. *Proceedings of the 12th International Conference Information Control Systems & Technologies (ICST 2024).* 2024. Vol. 3790. P. 99–109. URL: https://ceur-ws.org/Vol-3790/paper09.pdf **(Scopus)**

[7] Rozlomii I., Yarmilko A., Naumenko S. Security and efficiency models for cyber-physical systems in medical devices. *IEEE 19th International Conference on Computer Science and Information Technologies (CSIT).* 2024. P. 1–4. URL: https://doi.org/10.1109/CSIT65290.2024.10982678 **(Scopus)**

[8] Rozlomii I., Yarmilko A., Naumenko S. Innovative resource-saving security strategies for IoT devices. *Journal of Edge Computing.* 2025. Vol. 4, no. 1. P. 35–56. URL: https://doi.org/10.55056/jec.748 **(Scopus)**

[9] Rozlomii I., Yarmilko A., Naumenko S. Vulnerability modeling in cybersecurity of intelligent infrastructure networks. *International Scientific-Practical Conference.* 2024. P. 234–248. URL: https://doi.org/10.1007/978-3-031-90735-7_19 **(Scopus)**

[10] Rozlomii I., Yarmilko A., Naumenko S. Resource-efficient solutions for data security at the network level of the Medical Internet of Things. *Proceedings of the 7th International Conference on Informatics & Data-Driven Medicine (IDDM'2024).* 2024. P. 171–182. URL: https://ceur-ws.org/Vol-3892/paper13.pdf **(Scopus)**

[11] Faure E., Rozlomii I., Yarmilko A., Naumenko S. Protection of IoT networks: cryptographic solutions for cybersecurity management. *Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN 2024).* 2024. P. 24–34. URL: https://ceur-ws.org/Vol-3925/paper03.pdf **(Scopus)**

[12] Rozlomii I., Yarmilko A., Naumenko S. Integration of lightweight cryptography and artificial intelligence methods to increase the dependability of precision medicine systems. *Proceedings of the International Workshop on Computational Intelligence (IWSCI 2025), co-located with the IV International Scientific Symposium "Intelligent Solutions" (IntSol 2025), Kyiv–Uzhhorod, May 01–05, 2025.* 2025. P. 201–212. URL: https://ceur-ws.org/Vol-4035/Paper17.pdf **(Scopus)**

[13] Faure E., Rozlomii I., Naumenko S. Cryptographic load sharing method in critical infrastructure sensor networks. *Proceedings of the Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN-25).* 2025. P. 24–34. URL: https://ceur-ws.org/Vol-4024/paper01.pdf **(Scopus)**

[14] Rozlomii I., Naumenko S., Mykhailovskyi P., Lishchuk R. Methodology for selecting the protection strategy in IoT environments based on the device resource profile. *IEEE 6th KhPI Week on Advanced Technology (KhPIWeek).* 2025. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61436.2025.11288556 **(Scopus)**

[15] Danchenko O., Rozlomii I., Yarmilko A., Naumenko S. A lightweight Secure Boot mechanism for protecting the firmware of IoT devices. *Proceedings of the 13-th International Conference Information Control Systems & Technologies (ICST 2025), Odesa, Ukraine, September 24–26, 2025. CEUR Workshop Proceedings,* Vol. 4048. 2025. P. 240-250. URL: https://ceur-ws.org/Vol-4048/paper19.pdf **(Scopus)**

[16] Rozlomii I., Faure E., Yarmilko A., Naumenko S. The method for verifying firmware integrity in IoT devices for secure boot using lightweight hash functions. *Proceedings of the Cyber Security and Data Protection (CSDP 2025), Lviv,*

Ukraine, July 31, 2025. CEUR Workshop Proceedings, Vol. 4042. 2025. P. 105-116. URL: https://ceur-ws.org/Vol-4042/paper8.pdf **(Scopus)**

[17] Rozlomii I., Naumenko S., Trembovetskyi R. Method for rotating cryptographic keys based on time tokens for radio-electronic medical modules with limited resources. *Visnyk NTUU KPI Seriia-Radiotekhnika Radioaparatobuduvannia.* 2025. No. 102. P. 58–65. URL: https://doi.org/10.64915/RADAP.2025.102.58-65 **(Scopus)**

[18] Zabolotnii S., Rozlomii I., Yarmilko A., Naumenko S. Reconfigured CoARX architecture for implementing ARX hashing in microcontrollers of IoT systems with limited resources. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska.* 2025. Vol. 15, no. 4. P. 164–169. URL: https://doi.org/10.35784/iapgos.7782 **(Scopus)**

[19] Faure E., Rozlomii I., Naumenko S. Hybrid digital twin-driven anomaly detection in IoT telemetry using LSTM autoencoder. *Proceedings of the 2nd International Workshop on Data Analytics (WDA 2026), Kyiv, Ukraine, January 26, 2026.* CEUR Workshop Proceedings, Vol. 4155. 2025. P. 76–89. URL: https://ceur-ws.org/Vol-4155/paper06.pdf **(Scopus)**

[20] Розломій І. О., Косенюк Г. В., Науменко С. В., Михайловський П. В. Моделювання системи датчиків на базі мікроконтролера в ігровій симуляції «Смарт-будинок» з використанням шифрування. *Computer-Integrated Technologies: Education, Science, Production.* 2023. Вип. 53. С. 292–299. URL: https://doi.org/10.36910/6775-2524-0560-2023-53-43

[21] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Адаптивна криптографія для енергоефективного захисту пристроїв IoT. *Проблеми моделювання та автоматизації проектування.* 2024. № 1(19). С. 77–83. URL: https://doi.org/10.31474/2074-7888-2024-1-19-77-83

[22] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Модель безпеки взаємопов’язаних обчислювальних пристроїв на основі

полегшеної схеми шифрування для IoT. *Computer-Integrated Technologies: Education, Science, Production.* 2024. Вип. 55. С. 191–198. URL: https://doi.org/10.36910/6775-2524-0560-2024-55-24

[23] Rozlomii O., Yarmilko A., Naumenko S. The intelligent approaches to organizing secure information exchange in dynamic swarms of unmanned platforms. *Artificial Intelligence.* 2024. Vol. 29, no. 4. P. 151–158. URL: https://doi.org/10.15407/jai2024.04.151

[24] Розломій І. О., Науменко С. В. Моделювання взаємовпливу інформаційної безпеки та обчислювальних витрат у вбудованих пристроях. *Computer-Integrated Technologies: Education, Science, Production.* 2024. Вип. 57. С. 139–145. URL: https://doi.org/10.36910/6775-2524-0560-2024-57-16

[25] Розломій І. О., Фауре Е. В., Науменко С. В. Методи аутентифікації у вбудованих системах з обмеженими обчислювальними ресурсами. *Вимірювальна та обчислювальна техніка в технологічних процесах.* 2025. Вип. 81. С. 29–35. URL: https://doi.org/10.31891/2219-9365-2025-81-4

[26] Розломій І. О., Науменко С. В. Архітектура та функціональні особливості захищених систем керування базами даних нового покоління з підтримкою serverless та edge-обчислень. *Systems and Technologies.* 2025. №1 (69), С. 7–15. URL: https://doi.org/10.32782/2521-6643-2025-1-69.16

[27] Розломій І. О., Науменко С. В., Симонюк В. В., Пташенчук В. О., Зажома В. В. Полегшена криптографія для безпеки параметрів вібрації в постобробці 3D-друкованих деталей. *Інформаційні технології та суспільство.* 2025. № 2(17). С. 175–182. URL: https://doi.org/10.32689/maup.it.2025.2.25

[28] Rozlomii I., Koseniuk G., Naumenko S. Mechanisms for cryptographic code authentication control in sensor nodes with limited computing resources. *Computer-Integrated Technologies: Education, Science, Production.* 2025. No. 61. P. 193–198. URL: https://doi.org/10.36910/6775-2524-0560-2025-61-27

[29] Розломій І., Науменко С., Ковтюх В. Модель захищеного зберігання даних у розподілених базах даних на основі атрибутного шифрування для критичних інформаційно-комунікаційних систем. *Measuring and Computing Devices in Technological Processes.* 2026. № 1. С. 215–220. URL: https://doi.org/10.31891/2219-9365-2026-85-27

[30] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. Modern encryption methods in IoT: hardware solutions and cryptographic libraries for data. *Розвитки інформаційно-керуючих систем та технологій: монографія / за ред. В. Вичужаніна. Львів-Торунь: Liha-Pres,* 2024. P. 28–44. URL: https://doi.org/10.36059/978-966-397-422-4

[31] Розломій І. О., Ярмілко А. В., Науменко С. В. Ресурсоощадний підхід до побудови secure boot у вбудованих системах інтернету речей. *Системи контролю інформації та інтелектуальні технології. Досягнення та застосування: монографія / за ред. В. Вичужаніна. Львів-Торунь: Liha-Pres,* 2025. С. 102–122. URL: https://doi.org/10.36059/978-966-397-538-2-6

[32] Rozlomii I., Yarmilko A., Naumenko S. Towards distributed anomaly detection in smart networks: a power-efficient node-level approach. *Методи та засоби обчислювального інтелекту в управлінні смарт-системами: колективна монографія / за ред. В. М. Теслюка.* Львів, 2025. С. 109–118.

[33] Yarmilko A., Rozlomii I., Naumenko S. Robust communication clusters: Secure information exchange and redundant hashing for third-party inclusions localization. *КЗЯТПС-2023:* тези доп. Чернігів, 2023. P. 224–225.

[34] Науменко С. В., Розломій І. О. Information protection strategies in Industry 4.0: Encryption and cybersecurity for industrial systems. *Theoretical and Experimental Research in Materials Science and Mechanical Engineering:* матеріали IX Міжнар. наук.-практ. конф., Луцьк, 2023. Луцьк: Вежа-Друк, 2023. С. 191–193.

[35] Науменко С. В., Розломій І. О., Михайловський П. В. Забезпечення кібербезпеки в Smart-імплантах: роль полегшеної криптографії. *Інформаційна*

безпека та комп'ютерні технології: матеріали VII Міжнар. наук.-практ. конф., м. Кропивницький, 1 листоп. 2023 р. Кропивницький, 2023. С. 17–18.

[36] Rozlomii I. O., Yarmilko A., Naumenko S. Optimized hash functions for integrity control and data recovery in embedded systems. *Information-Management Systems and Technologies:* матеріали XI Міжнар. наук. конф. 2023. С. 31–34.

[37] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The comprehensive IoT security strategy using hardware and software encryption methods. *Information-Management Systems and Technologies:* матеріали XII Міжнар. наук. конф., Одеса, 23 – 25 верес. 2024 р. Одеса, 2024. С. 63–65.

[38] Розломій І. О., Ярмілко А. В., Науменко С. В. Інтелектуальні підходи до організації інформаційного обміну в динамічних зграях безпілотних платформ. *Штучний інтелект та інтелектуальні системи (AIIS'2024):* матеріали XXIV Міжнар. наук.-техн. конф. 2024. С. 144–149.

[39] Науменко С. В., Михайловський П. В., Стабецька Т. А. Сучасні технології захисту даних у вбудованих пристроях з обмеженими ресурсами. *Free and Open Source Software:* матеріали Міжнар. наук.-практ. конф., Харків, 13 – 14 лют. 2025 р. Харків, 2025. С. 63–64.

[40] Розломій І. О., Науменко С. В. Конфігурований ARX-примітив для енергоефективного хешування у пристроях IoT. *Інформаційні системи та технології: результати і перспективи (IST 2025):* матеріали II Міжнар. наук.-практ. конф. 2025. С. 269–271.

[41] Чікін Д. М., Науменко С. В., Розломій І. О. Захист персональних даних в IoT-пристроях із застосуванням штучного інтелекту. *Інформаційна безпека та комп'ютерні технології:* тези доп. VIII Міжнар. наук.-практ. конф., м. Кропивницький, 24 – 25 квіт. 2025 р. Кропивницький: ЦНТУ, 2025. С. 12–13.

[42] Розломій І. О., Науменко С. В., Михайловський П. В. Формування сеансових ключів у сенсорних пристроях з обмеженим обсягом пам'яті на основі часових токенів. *Комплексне забезпечення якості технологічних процесів та*

систем (КЗЯТПС – 2025): матеріали XV Міжнар. наук.-практ. конф., Чернігів, 22 – 23 трав. 2025 р. Чернігів: НУ «Чернігівська політехніка», 2025. Вип. 2. С. 248–249.

[43] Розломій І. О., Науменко С. В. Метод розподілу криптографічного навантаження між мікроконтролерами в edge-архітектурах на основі енергетичної моделі. *Проблеми комп'ютерних наук, програмного моделювання та безпеки цифрових систем:* матеріали II Міжнар. наук.-практ. конф., Луцьк, 2025. Луцьк: ЛНТУ, 2025. С. 112–115.

[44] Розломій І. О., Науменко С. В. Модель адаптивної побудови довірчих IoT-мереж із динамічною ротацією вузлів. *Програмне та апаратне забезпечення в інформаційних технологіях:* матеріали Міжнар. наук.-практ. конф. молодих вчених та студентів, Луцьк, 6 трав. 2025 р. / відп. ред. Т. В. Терлецький. Луцьк: ЛНТУ, 2025. Вип. 1. С. 136–139.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

КФС – кіберфізична система;

AES – Advanced Encryption Standard;

ARX – Addition-Rotation-XOR;

XTEA – eXtended Tiny Encryption Algorithm;

IoT – Internet of Things;

RoT – Root of Trust;

SPI – Serial Peripheral Interface;

USART – Universal Synchronous/Asynchronous Receiver/Transmitter;

ECDSA – Elliptic Curve Digital Signature Algorithm;

ECDH – Elliptic-Curve Diffie–Hellman;

ОЗП – оперативний запам'ятовувальний пристрій;

K12 – KangarooTwelve;

ROM – Read-Only Memory;

RAM – Random Access Memory;

SRAM – Static Random Access Memory;

SHA-256/SHA-2/SHA-3 – Secure Hash Algorithm 256-bit/2-bit/3-bit;

SWD – Serial Wire Debug;

JTAG – Joint Test Action Group;

UART – Universal Asynchronous Receiver-Transmitter;

OTA – Over-the-Air;

TPM – Trusted Platform Module;

HSM – Hardware Security Module;

EEPROM – Electrically Erasable Programmable Read-Only Memory;

GPIO – General-Purpose Input/Output;

LED – light-emitting diode;

RSA-2048 – Rivest, Shamir, Adleman-2048;

API – Application Programming Interface;

ISO – International Organization for Standardization;

NIST – National Institute of Standards and Technology;

SIMD – Single Instruction Multiple Data;

AEAD – authenticated encryption with associated data;

MAC/HMAC – Message Authentication Code/Hash-based Message Authentication Code;

TRNG – True Random Number Generator;

OTP – One-Time Programmable;

RDP – Read Protection Level;

TEE – Trusted Execution Environment;

PoR – Proof of Retrievability;

PoE – Proof of Execution;

REST – Representational State Transfer;

EDHOC – Ephemeral Diffie-Hellman Over COSE;

OSCORE – Object Security for Constrained RESTful Environments;

TLS/DTLS – Transport Layer Security/Datagram Transport Layer Security;

LEA – Lightweight Encryption Algorithm;

RTOS – Real-Time Operating System;

MCU – Multipoint control unit;

ABE – Attribute-based encryption;

SGX – Software Guard Extensions;

SDK – Software Development Kit;

PBE – Public Beta Environment;

CP-ABE – Ciphertext-Policy Attribute-Based Encryption;

KP-ABE – Key-Policy Attribute-Based Encryption.

ВСТУП

Актуальність теми дослідження. Широке впровадження кіберфізичних систем (КФС) у промисловості, медицині та інших галузях супроводжується зростанням обсягів інформації, що функціонує в їх межах [1]. Значна частина цих даних є критичною або конфіденційною. Сенсорні вузли, мікроконтролери, вбудовані пристрої та інші компоненти КФС дедалі частіше зазнають кібератак, спрямованих на перехоплення, модифікацію чи підміну даних [2], [3]. Це особливо небезпечно в умовах обмежених ресурсів, коли традиційні засоби захисту є надто складними або спричиняють неприйнятні обчислювальні й енергетичні витрати.

Сучасні виклики у сфері інформаційної безпеки КФС вимагають не лише забезпечення конфіденційності, цілісності й автентичності даних, а й адаптації методів захисту до ресурсних обмежень середовища [4]–[7]. У практичних умовах важливими є не лише рівень захисту, а й час виконання криптографічних операцій, витрати пам’яті та енергоспоживання [8]–[11]. Тому дослідження полегшених криптографічних моделей і оцінювання їх ефективності для вбудованих пристроїв КФС є актуальним.

Проблема ускладнюється тим, що обчислювальні модулі КФС часто виконують кілька задач паралельно, зокрема обробку сигналів, комунікацію та прийняття рішень у реальному часі [12]–[15]. Впровадження криптографічних механізмів має здійснюватися з мінімальним впливом на інші процеси [16], [17], що потребує розробки оптимізованих моделей захисту. Актуальність посилюється зростанням кількості атак на безпілотні системи [18]–[20], медичні пристрої [21]–[23], енергетичні мережі [24]–[26] та інші критичні інфраструктури, що базуються на КФС.

Попри наявність численних досліджень у сфері криптографії, більшість з них орієнтовані на використання в середовищах із відносно високим рівнем ресурсів. Недостатньо опрацьованими залишаються питання комплексного

криптографічного захисту на рівні сенсорних вузлів і мікроконтролерів із дуже обмеженими можливостями. Актуальними залишаються задачі побудови моделей вибору та розподілу криптографічного навантаження, розробки механізмів захищеного завантаження (Secure Boot), генерації ключів у режимах обмеженої пам’яті, а також створення моделей обміну за захищеними каналами між компонентами КФС.

У наукових дослідженнях, присвячених захисту інформації в сенсорних і розподілених обчислювальних мережах, значна увага приділяється криптографічним механізмам, адаптованим до обмежених апаратних і енергетичних ресурсів. Зокрема, у працях A. K. Singh, D. E. Boubiche [27], O. A. Khashan, R. Ahmad [28] досліджено базові загрози безпеці сенсорних мереж, запропоновано моделі захищеного обміну даними та криптографічні схеми з низькими обчислювальними витратами. У роботах P. Kumar [29], Q.Wang [30], H. Li [31] розглядаються протоколи автентифікації та управління ключами для вузлів із низькою продуктивністю, проте зазначені підходи переважно орієнтовані на стабільні режими живлення та не враховують динамічні умови функціонування вузлів.

Аспекти впровадження криптографічних протоколів у вбудованих пристроях і системах реального часу розкриті в працях J. Babatope [32], Z. Zhang [33], X. Wang, Q. Hao, C. Silva [34], де аналізуються загальні принципи побудови стійких схем автентифікації та захисту каналів зв’язку. Водночас більшість запропонованих рішень розраховані на середовища з відносно високим рівнем обчислювальних ресурсів і не забезпечують достатньої адаптації до умов обмеженої пам’яті, переривчастого живлення та жорстких часових обмежень, характерних для компонентів кіберфізичних систем [35].

Окремий напрям досліджень становлять роботи, присвячені полегшеним криптографічним алгоритмам [36]–[39]. У працях Y. Chen, Z. Bao [40], L. Sleem [41], V. A. Thakor, V. N. Kumar здійснено детальний аналіз таких алгоритмів, як

SPECK, PRESENT, LEA, SIMON та їхніх модифікацій, з точки зору швидкодії, енергоспоживання та апаратної реалізації [42]. Разом із тим у зазначених дослідженнях основна увага приділяється порівнянню окремих алгоритмів, тоді як питання обґрунтованого вибору криптографічних механізмів залежно від ролі конкретного пристрою в системі, його режиму роботи та доступних ресурсів залишаються недостатньо опрацьованими.

У працях, присвячених організації захищеного обміну даними між вузлами розподілених систем, зокрема у дослідженнях A. Al-Haj [43], A. Allakany [44], S. Singh [45], R. Vinoth [46], Y. Zhang, K. Cheng [47], запропоновано підходи до побудови захищених каналів зв’язку та схем взаємної автентифікації. Однак більшість таких рішень орієнтовані на мережі з централізованим керуванням або високим рівнем синхронізації між вузлами та не враховують особливості децентралізованих кіберфізичних систем із змінною топологією та автономною роботою компонентів.

Вітчизняні дослідники, зокрема А.А. Коваленко [48], І.Є. Андрущак [49], В.В. Поліщук [50], Б.Ю. Жураковський [51] у своїх наукових працях розглядали актуальні аспекти захисту IoT-мереж, включаючи методи виявлення аномального мережевого трафіку, підходи до проєктування систем інформаційної безпеки для IoT-інфраструктур, розвиток квантово-стійких криптографічних алгоритмів, а також методи побудови захищених комунікаційних каналів у мережах нового покоління. Разом з тим проблеми інтеграції полегшених криптографічних механізмів у компоненти кіберфізичних систем з урахуванням обмежених ресурсів, динаміки взаємодії вузлів і реальних умов експлуатації залишаються актуальними та потребують подальшого комплексного дослідження.

Разом з тим, недостатньо досліджено питання криптографічного захисту в реальних умовах роботи сенсорних і виконавчих модулів, які функціонують автономно, з переривчастим живленням, змінними топологіями мережі, та часто – в агресивних середовищах. Крім того, актуальним залишається завдання

побудови узагальненої моделі інформаційної безпеки на рівні компонентів КФС, яка дозволяла б не лише впровадити полегшені методи захисту, а й оцінити їх доцільність для конкретного типу пристрою та його ролі в системі.

У дисертаційній роботі вирішується важлива науково-технічна задача побудови методу та моделей криптографічного захисту інформації у компонентах кіберфізичних систем з урахуванням ресурсних обмежень. Дана задача передбачає удосконалення існуючих підходів до захищеного завантаження, вибору та адаптації криптографічних механізмів, а також побудови захищеного обміну між компонентами КФС з урахуванням динаміки їх взаємодії та обмеженого доступу до енергетичних і обчислювальних ресурсів.

Зв’язок роботи з науковими програмами, планами, темами. Дослідження, результати яких представлено в дисертаційній роботі, відповідають пріоритетному напряму розвитку науки і техніки України «Інформаційні та комунікаційні технології» та його тематичному напряму «Кіберфізичні системи. Інтернет речей. Робототехніка» і виконувалися відповідно до програм і планів науково-дослідних робіт Черкаського національного університету імені Богдана Хмельницького, у тому числі в рамках науководослідної теми «Технічне і програмне забезпечення новітніх комп'ютерних інформаційних систем» (номер державної реєстрації 0126U001916), у якій автор брав участь як виконавець.

Метою дисертаційної роботи є підвищення ефективності захисту інформації у компонентах кіберфізичних систем шляхом розробки методу та моделей, адаптованих до умов обмежених обчислювальних ресурсів, з урахуванням особливостей обміну даними та захищеного завантаження в архітектурах типу edge/fog.

Для досягнення поставленої мети необхідно вирішити такі задачі:

1. Удосконалити модель захисту інформації на рівні вбудованих пристроїв та сенсорних модулів, яка включає механізми захищеного

завантаження, вибору та інтеграції криптографічних алгоритмів, а також управління ключами.

2. Побудувати модель захисту інформаційного обміну між компонентами КФС, включаючи протидію атакам на канали передавання, механізми шифрування, автентифікації та захисту від повторної передачі.

3. Удосконалити модель динамічного розподілу криптографічного навантаження між компонентами кіберфізичної системи та периферійними обчисленнями (edge/fog) з урахуванням стану ресурсів, характеристик каналів зв’язку та вимог до безпеки інформаційного обміну.

4. Розвинути метод побудови комплексної системи криптографічного захисту інформації у компонентах кіберфізичних систем на основі інтеграції механізмів захищеного завантаження, адаптивного вибору полегшених криптографічних алгоритмів та організації захищеного інформаційного обміну з урахуванням обмежених обчислювальних ресурсів.

5. Оцінити ефективність запропонованого методу у програмно-апаратному середовищі з використанням мікроконтролера STM32 за визначеними метриками та порівняння з існуючими криптографічними рішеннями.

Об’єктом дослідження є процеси криптографічного захисту інформації у компонентах кіберфізичних систем, функціонування яких відбувається в умовах обмежених обчислювальних ресурсів.

Предметом дослідження є методи, моделі та алгоритми криптографічного захисту інформації, що реалізуються на рівні сенсорних модулів, вбудованих пристроїв та каналів взаємодії між компонентами кіберфізичних систем.

Методи досліджень. Для вирішення задачі аналізу архітектур кіберфізичних систем, класифікації загроз та вибору критеріїв ефективності криптографічного захисту використовувалися методи: теорії інформації, теорії систем та системного аналізу, ризик-орієнтованого моделювання.

Для вирішення задачі розробки моделей криптографічного захисту інформації на рівні сенсорних вузлів і вбудованих пристроїв, включаючи захищене завантаження програмного коду, використовувалися методи: формальних мов і автоматів, дискретної математики, теорії графів, математичної логіки, апаратно-програмного моделювання.

Для вирішення задачі побудови моделі захисту інформаційного обміну між компонентами КФС та організації протидії атакам на канали передавання інформації використовувалися методи: криптографічного аналізу, теорії мережевої безпеки, протоколів автентифікації, аналізу загроз і векторів атак.

Для вирішення задачі розробки методу побудови комплексної системи криптографічного захисту інформації у КФС з урахуванням обмежених обчислювальних ресурсів використовувалися методи: комбінаторної оптимізації, математичного моделювання, теорії алгоритмів, аналізу енергоспоживання та продуктивності систем реального часу.

Для вирішення задачі оцінювання ефективності запропонованого методу та моделей використовувалися методи: експериментального дослідження, статистичного аналізу, теорії ймовірності і математичної статистики, а також методи об’єктно-орієнтованого програмування при реалізації програмного забезпечення для STM32 у середовищі FreeRTOS.

Наукова новизна отриманих результатів:

- **вперше побудовано** модель криптографічного захисту інформаційного обміну в середовищі взаємодії компонентів КФС, яка за рахунок використання полегшених протоколів встановлення захищеного каналу та механізмів захисту від атак повторної передачі забезпечує цілісність і конфіденційність даних у децентралізованих архітектурах типу edge/fog з урахуванням динамічної топології та ресурсних обмежень вузлів;
- **удосконалено** модель захисту інформації на рівні сенсорних модулів та вбудованих пристроїв, яка, на відміну від існуючих підходів, базується на

поєднанні архітектури захищеного завантаження Secure Boot, контролю цілісності програмного коду та механізмів керування криптографічними ключами з урахуванням обмежених обчислювальних ресурсів, що забезпечує підвищену стійкість до атак на етапі ініціалізації пристрою;

– **удосконалено** модель динамічного розподілу криптографічного навантаження між компонентами кіберфізичної системи та периферійними обчисленнями, яка на основі узагальненої функції вартості та обмежень за енергоспоживанням, затримками і доступними ресурсами забезпечує адаптивний вибір локального або делегованого виконання криптографічних операцій;

– **набув подальшого розвитку** метод побудови комплексної системи криптографічного захисту інформації у компонентах КФС, який шляхом інтеграції захищеного завантаження, вибору полегшених криптографічних алгоритмів та безпечного інформаційного обміну забезпечує захист в умовах обмежених обчислювальних ресурсів.

Практичне значення отриманих результатів:

1. Розроблено алгоритм перевірки цілісності та автентичності програмного коду у процесі завантаження на основі архітектури Secure Boot для мікроконтролерів STM32 та ESP32. Експериментальні випробування показали, що перевірка автентичності виконується за час до 120 мс при обсязі коду до 128 КБ, використовуючи до 5 КБ оперативної пам’яті.

2. Розроблено алгоритм адаптивного вибору криптографічного захисту з урахуванням характеристик пристрою, ресурсоємності алгоритму, рівня криптостійкості та специфіки функціонування вузла. У результаті реалізації алгоритму на основі мікроконтролера STM32F407 досягнуто зниження енергоспоживання криптографічних операцій у середньому на 23% порівняно з базовим варіантом використання AES-128.

3. Розроблено алгоритм захищеного обміну інформацією між компонентами КФС в архітектурі edge/fog, який враховує обмеження на час

обміну, динамічну топологію та можливість атак повторної передачі. Тестування у мережі з 20 вузлів показало зменшення затримки до 1,4 мс та стабільність з'єднання в умовах зміни маршрутизації.

4. Розроблено алгоритм реалізації методу побудови комплексної системи криптографічного захисту інформації у гетерогенному середовищі КФС, який визначає послідовність ініціалізації довіри, узгодження криптографічних параметрів, встановлення захищених сесій та динамічного розподілу криптографічного навантаження між сенсорними вузлами, вбудованими пристроями, edge-компонентами та хмарною інфраструктурою з урахуванням обмежень обчислювальних ресурсів.

5. Створено імітаційну модель методу, реалізованого на базі мікроконтролерів STM32 та ESP32, з використанням середовища Renode, FreeRTOS і засобів моніторингу споживання енергії. У ході тестування підтверджено зниження обчислювального навантаження на 35–40% та підвищення загальної продуктивності на 20–25% у порівнянні з типовими реалізаціями криптографічного захисту.

Особистий внесок здобувача. Дисертація є самостійно виконаною завершеною роботою здобувача. Наукові результати і практичні розробки, що містяться в дисертаційній роботі, отримані автором самостійно.

У роботах, опублікованих у співавторстві, автором: [1], [5], [11], [15], [16], [21], [25], [28], [36], [39] – розроблено та обґрунтовано методи використання полегшених криптографічних і хеш-функцій для забезпечення цілісності, автентичності та конфіденційності даних у вбудованих пристроях і сенсорних вузлах з обмеженими ресурсами; виконано аналіз ефективності запропонованих рішень для мікроконтролерних платформ; [2], [4], [7], [8], [12], [14], [22], [24], [30], [32], [34], [35] – виконано дослідження архітектур кіберфізичних та IoT-систем, сформульовано вимоги до полегшених криптографічних механізмів, запропоновано моделі захисту інформації з урахуванням енергоспоживання,

обчислювальних витрат і ролі пристрою в системі; [8], [13], [14], [21], [22], [24], [43], [44] – запропоновано та узагальнено моделі динамічного вибору й розподілу криптографічного навантаження між компонентами кіберфізичних систем, edge- та fog-вузлами; розроблено критерії ефективності та функції вартості для адаптивного прийняття рішень; [6], [18], [27], [36], [40] – здобувачем виконано модифікацію та аналіз криптографічних примітивів і блокових перетворень, орієнтованих на реалізацію в середовищі з обмеженими апаратними ресурсами; оцінено їх вплив на швидкодію, пам’ять і стійкість до атак; [15], [16], [28], [31], [36] – розроблено методи захищеного завантаження (Secure Boot) та контролю цілісності програмного коду для вбудованих пристроїв; визначено структуру алгоритмів, механізми перевірки автентичності та сценарії їх практичного застосування; [17], [42] – запропоновано методи формування та ротації криптографічних ключів у сенсорних і радіоелектронних модулях на основі часових токенів, адаптовані до умов обмеженої пам’яті та нестабільного живлення; [23], [33], [37], [38], [44] – розроблено підходи до організації захищеного інформаційного обміну в динамічних децентралізованих кіберфізичних середовищах, зокрема у зграях безпілотних платформ, із урахуванням змінної топології та автономності вузлів; [3], [9], [12], [19], [26], [29], [32], [41] – виконано дослідження суміжних аспектів підвищення надійності та безпеки кіберфізичних і хмарних систем, зокрема із застосуванням інтелектуальних та адаптивних підходів; отримані результати використано для розширення та узагальнення моделей, запропонованих у дисертації.

Усі результати, що використані в дисертаційній роботі з публікацій, виконаних у співавторстві, отримані здобувачем особисто або за його безпосередньої участі, що відповідає тематиці та науковим завданням дисертаційного дослідження.

Апробація результатів дисертації. Основні результати дисертаційної роботи доповідалися та обговорювалися на:

- XIII міжнародній науково-практичній конференції «Комплексне забезпечення якості технологічних процесів та систем» (м. Чернігів, 26 травня 2023 р.);
- 11-th International Conference «Information Control Systems & Technologies» (Odesa, Ukraine, September 21–23, 2023);
- VII міжнародній науково-практичній конференції «Інформаційна безпека та комп'ютерні технології» (м. Кропивницький, 1 листопада 2023);
- 8th International Conference «Mathematical Modeling and Simulation Systems» (MODS2023) (Chernihiv, Ukraine, November 13–15, 2023);
- 6th International Conference on Informatics & Data-Driven Medicine (Bratislava, Slovakia, November 17–19, 2023);
- 3rd International Workshop on Information Technologies: Theoretical and Applied Problems (Ternopil, Ukraine, Opole, Poland, November 22–24, 2023);
- 4th Edge Computing Workshop (Zhytomyr, Ukraine, April 5, 2024);
- 4th IEEE International Conference on Smart Information Systems and Technologies (Astana, Kazakhstan, May 15–17, 2024);
- 14th International Conference on Advanced Computer Information Technologies (České Budějovice, September 19–21 2024);
- XII International Scientific Conference «Information-Management Systems and Technologies» (Odesa, 23–25 September 2024);
- 5th KhPI Week on Advanced Technology (Kharkiv, Ukraine, October 7–11, 2024);
- 14th International Conference on Dependable Systems, Services and Technologies (Athens, Greece, October 11–13, 2024);
- XXIV міжнародній науково-технічній конференції «Штучний інтелект та інтелектуальні системи- AIIS'2024» (м. Київ, 18–19 жовтня 2024 р.);

- 19th International Conference on Computer Science and Information Technologies (Lviv, Ukraine, 16–19 October 2024);
- 3-rd International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN 2024) (Kyiv, Ukraine, January 24–27, 2024);
- XVI міжнародній науково-практичній конференції «Free and Open Source Software» (м. Харків, 13–14 лютого 2025 р.);
- VIII міжнародній науково-практичній конференції «Інформаційна безпека та комп'ютерні технології» (м. Кропивницький, 24–25 квітня 2025 р.);
- II міжнародній науково-практичній конференції «Проблеми комп'ютерних наук, програмного моделювання та безпеки цифрових систем» (м. Луцьк, 9–11 червня 2025 р.);
- міжнародній науково-практичній конференції «Програмне та апаратне забезпечення в інформаційних технологіях» (м. Луцьк, 6 травня 2025 р.);
- 2nd International Workshop on Data Analytics (Kyiv, Ukraine, January 26, 2026).

Публікації. Результати дослідження опубліковані у 44 наукових роботах, в тому числі 19 наукові статті, що входять до наукометричних баз даних Scopus та / або Web of Science [1] – [19], 10 статей у наукових виданнях, що входять до переліку МОН України [20] – [29], 3 розділах колективних монографій [30] – [32], 12 доповідях на науково-практичних конференціях [33] – [44].

Структура та обсяг дисертаційної роботи. Дисертаційна робота складається з вступу, чотирьох розділів, висновків, списку використаних джерел і додатків. Загальний обсяг роботи становить 229 сторінок. Робота містить 27 таблиць та 49 рисунків та 178 найменувань у списку використаних джерел.

1. АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ПІДХОДІВ ДО ЗАХИСТУ ІНФОРМАЦІЇ В КІБЕРФІЗИЧНИХ СИСТЕМАХ. ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1. Структура кіберфізичних систем

Кіберфізичні системи – інтегровані комплекси, що поєднують фізичні процеси, обчислювальні модулі та комунікаційні технології у єдину керовану структуру [52]. Їх робота базується на безперервній взаємодії між сенсорними вузлами, виконавчими механізмами, вбудованими контролерами, мережевою інфраструктурою та сервісними рівнями, які забезпечують аналіз, прийняття рішень і віддалене управління [53]–[55]. Завдяки такому поєднанню КФС здатні не лише сприймати дані про стан об’єкта чи середовища, а й реагувати на зміни у реальному часі, що робить їх ключовим елементом сучасних технологічних платформ.

Сфери застосування КФС охоплюють промислову автоматизацію [56], робототехніку [57], енергетичні мережі [58], транспортні системи [59], безпілотні платформи [60], медичні пристрої [61], «розумну» інфраструктуру та системи моніторингу [62]. У таких галузях КФС виконують функції адаптивного керування, виявлення аномалій, збирання та аналітики даних, дистанційного управління та підвищення безпеки технологічних процесів [63]–[65]. Критичність виконуваних функцій зумовлює необхідність коректної побудови архітектури КФС та забезпечення стійкої взаємодії компонентів у середовищі з обмеженими ресурсами [66].

Попри різноманітність застосувань, КФС мають типову узагальнену архітектуру, яка включає кілька взаємопов’язаних рівнів [67]. Кожен рівень виконує власні функції, взаємодіє з попереднім та наступним рівнями та формує завершений контур збирання, обробки, передавання й використання інформації. У цьому дослідження використовується семирівнева модель КФС, яка дозволяє

структуровано представити систему та окреслити області застосування полегшених криптографічних механізмів.

Узагальнення архітектури до семи рівнів дає змогу чітко відокремити фізичні, обчислювальні, мережеві та сервісні компоненти, а також локалізувати області виникнення загроз і точки застосування криптографічного захисту [68], [69]. Такий підхід забезпечує можливість аналізу інформаційних потоків на всіх етапах життєвого циклу даних – від первинного вимірювання до відображення результатів оператору чи зовнішнім інформаційним системам. Нижче наведено характеристику кожного рівня узагальненої архітектури КФС:

1. Рівень сенсорів забезпечує первинну взаємодію КФС з фізичним середовищем, перетворюючи фізичні величини на цифрові дані. На цьому рівні формуються потоки вимірювальної інформації, які надалі використовуються для аналізу стану об’єкта, виявлення аномалій та вироблення керуючих дій [70]. Сенсорні вузли можуть бути як простими однофункціональними давачами, так і складними комбінованими модулями з вбудованими мікроконтролерами ультранизького енергоспоживання.

Сенсори часто працюють в умовах обмеженого живлення, що зумовлює жорсткі вимоги до енергетичної ефективності як вимірювальних, так і криптографічних операцій [71], [72]. У частині попередньої обробки тут можуть виконуватися прості перетворення сигналів, фільтрація шумів, усереднення та грубе агрегування даних, якщо це допускають ресурси мікроконтролера [73]. Ключові характеристики сенсорного рівня узагальнено в табл. 1.1.

2. Рівень актуаторів забезпечує перетворення цифрових команд, сформованих вбудованими контролерами чи edge-пристроями, у фізичні дії щодо об’єкта керування [74]. Саме на цьому рівні відбувається вплив на технологічний процес, механічні вузли, медичні органи-мішені чи інші фізичні об’єкти.

Виконавчі механізми можуть реалізовувати широкий спектр дій: від простого вмикання й вимикання до точного дозування, позиціонування,

електростимуляції або оптичного впливу [75]. Для критичних застосувань (медицина, транспорт, енергетика) важливим є коректне інтерпретування команд та неможливість їхнього довільного підміни або блокування. Тому поряд зі стандартними механізмами контролю положення та зворотного зв’язку актуатори мають інтегруватися у контур криптографічного захисту команд керування, отримуючи їх тільки з автентичних джерел.

Таблиця 1.1 – Основні характеристики сенсорного рівня КФС

Характеристика	Опис
Функціональне призначення	Первинна взаємодія з фізичним середовищем, перетворення фізичних величин у цифрові дані
Типи пристроїв	Сенсорні вузли, давачі, комбіновані сенсорні модулі з вбудованими мікроконтролерами
Обчислювальні ресурси	Ультранизька обчислювальна потужність, обмежений обсяг RAM і Flash-пам’яті
Джерело живлення	Батарейне, енергозбирання, нестабільне або обмежене живлення
Типові операції обробки	Попередня фільтрація сигналів, усереднення, базове агрегування даних
Обмеження ресурсів	Жорсткі обмеження на енергоспоживання, час виконання та пам’ять
Комунікаційні інтерфейси	SPI, I2C, UART, однопровідні або бездротові інтерфейси малого радіусу дії
Типові загрози	Фізичний доступ, підміна сенсорних даних, витік ключового матеріалу
Криптографічні вимоги	Мінімальні обчислювальні витрати, використання полегшених механізмів хешування та автентифікації
Роль у архітектурі КФС	Джерело первинних даних для всіх наступних рівнів системи

3. Рівень вбудованого контролера забезпечує перехід від «сирих» сенсорних даних до керуючих рішень у реальному часі. Це ядро фізичної частини КФС, де виконуються алгоритми обробки сигналів, регулювання, діагностики та локального реагування на події. На цьому ж рівні розміщуються базові механізми криптографічного захисту, включно з архітектурою Secure Boot, перевіркою

цілісності коду, автентифікацією підключених модулів та полегшеними протоколами обміну [76].

Вбудовані контролери працюють під керуванням спеціалізованих операційних систем реального часу (FreeRTOS, Zephyr тощо), які забезпечують планування задач [77], [78], обробку переривань та взаємодію з периферією через інтерфейси SPI, I2C, UART, CAN та інші шини [79]–[82]. В умовах обмежених ресурсів саме на цьому рівні часто приймається рішення про розподіл криптографічного навантаження між локальними та периферійними обчисленнями. Основні атрибути рівня вбудованого контролера подано в табл. 1.2.

Таблиця 1.2 – Характеристики рівня вбудованого контролера

Характеристика	Опис
Функціональне призначення	Локальна обробка сенсорних даних, формування керуючих рішень у реальному часі
Типові апаратні платформи	Мікроконтролери класу ARM Cortex-M, RISC-V, енергоефективні SoC
Обчислювальні ресурси	Обмежена продуктивність CPU, фіксований обсяг RAM та Flash-пам’яті
Програмне середовище	Операційні системи реального часу (FreeRTOS, Zephyr) або bare-metal
Типові функції	Обробка сигналів, регулювання, діагностика, реакція на події
Периферійні інтерфейси	SPI, I2C, UART, CAN, GPIO, ADC, таймери
Криптографічні механізми	Secure Boot, перевірка цілісності коду, автентифікація модулів, полегшені алгоритми шифрування
Розподіл криптографічного навантаження	Часткове виконання криптографічних операцій локально з можливістю делегування на edge-рівень
Типові загрози	Підміна прошивки, ін’єкція шкідливого коду, компрометація пам’яті
Роль у архітектурі КФС	Ключовий елемент керування та прийняття рішень у фізичному контурі системи

4. Рівень мережевої взаємодії забезпечує передавання даних між компонентами КФС, організацію маршрутизації, кластеризації та керування якістю обслуговування трафіку [83]. На цьому рівні формуються мережі різної топології – від простих зіркоподібних конфігурацій до складних mesh-структур із динамічною зміною маршрутування [84]. Мережеві шлюзи та бездротові модулі реалізують транспорт даних через Wi-Fi, BLE, LoRaWAN, Zigbee, 6LoWPAN й інші технології, а також виконують протоколи захищеного обміну [85]–[88]. Саме тут інтегруються полегшені варіанти протоколів TLS, протоколи EDHOC, OSCORE [89], [90], механізми полегшеного хешування та узгодження ключів. Для критичних застосувань важливими є керування затримками, пріоритизація трафіку та стійкість до атак на канали зв’язку, включаючи перехоплення, підміну й повторну передачу пакетів. Узагальнена характеристика мережевого рівня наведена в табл. 1.3.

Таблиця 1.3 – Атрибути мережевого рівня КФС

Характеристика	Опис
Функціональне призначення	Передавання даних між компонентами КФС, організація мережевої взаємодії
Типові мережеві технології	Wi-Fi, BLE, Zigbee, LoRaWAN, 6LoWPAN, дротові промислові шини
Топології мережі	Зіркоподібна, деревоподібна, mesh, гібридна з динамічною маршрутизацією
Мережеві компоненти	Бездротові модулі, мережеві шлюзи, маршрутизатори, комунікаційні контролери
Протоколи обміну	CoAP, MQTT, UDP/TCP, полегшені варіанти TLS/DTLS
Криптографічні механізми	Полегшені протоколи встановлення захищених з’єднань, хешування, узгодження ключів
Керування трафіком	Пріоритизація повідомлень, керування затримками, контроль якості обслуговування
Типові загрози	Перехоплення трафіку, підміна пакетів, повторна передача (replay-атаки)
Вимоги до безпеки	Автентичність вузлів, цілісність і конфіденційність переданих даних
Роль у архітектурі КФС	Забезпечення зв’язності між фізичними, edge- та хмарними рівнями

5. Рівень периферійних обчислень (edge-рівень) забезпечує локальну обробку та аналіз даних поблизу місця їхнього виникнення [91]. Це проміжна ланка між вбудованими контролерами та хмарними сервісами, яка дозволяє скоротити затримки, зменшити обсяг переданих даних та частково розвантажити обчислювальні ресурси хмари. Основні характеристики edge-рівня зведено в табл. 1.4.

Таблиця 1.4 – Характеристика периферійного (edge) рівня

Характеристика	Опис
Функціональне призначення	Локальна обробка та аналіз даних поблизу джерела їх виникнення
Типові пристрої	Одноплатні комп’ютери, edge-шлюзи, спеціалізовані обчислювальні вузли
Обчислювальні ресурси	Помірна або підвищена продуктивність CPU, більший обсяг RAM і постійної пам’яті порівняно з мікроконтролерами
Програмне середовище	Вбудовані дистрибутиви Linux, контейнеризовані середовища, edge-фреймворки
Типові функції	Агрегування даних, попередній аналіз, виявлення аномалій, локальне прийняття рішень
Взаємодія з іншими рівнями	Збір даних з сенсорних і мережевих рівнів, передавання узагальненої інформації до хмари
Криптографічні механізми	Використання асиметричної криптографії, складніших протоколів встановлення ключів, керування сеансами
Розподіл криптографічного навантаження	Перенесення ресурсоємних криптографічних операцій з нижчих рівнів
Типові загрози	Компрометація вузла, атаки на програмні сервіси, підміна агрегованих даних
Роль у архітектурі КФС	Проміжна ланка між фізичними компонентами та хмарною інфраструктурою

Edge-пристрої (одноплатні комп’ютери, спеціалізовані шлюзи) виконують задачі агрегування потоків від багатьох сенсорних вузлів, попереднього аналізу, виявлення аномалій, локального прийняття рішень і розподілу криптографічного

навантаження [92]. У частині захисту на цьому рівні можливе використання більш ресурсомістких алгоритмів, включно з асиметричною криптографією та складнішими протоколами встановлення ключів, оскільки ресурси edge-вузлів, як правило, суттєво перевищують можливості окремих сенсорів та мікроконтролерів [93]–[95].

6. Рівень хмарних сервісів забезпечує централізоване зберігання, масштабовану обробку та довготривалу аналітику даних, що надходять з периферійних та мережевих рівнів. Хмарна інфраструктура виконує задачі агрегування історичних даних, побудови моделей машинного навчання, прогнозування станів системи та формування політик для підлеглих рівнів [96], [97].

У контексті криптографічного захисту саме тут можуть розгортатися повнофункціональні засоби керування ключами, централізовані служби автентифікації, системи виявлення вторгнень та платформи для аналізу журналів безпеки. Окрім цього, на хмарному рівні реалізуються механізми віддаленого оновлення прошивок (OTA-оновлення), що вимагає побудови надійних схем перевірки цілісності та автентичності коду перед його завантаженням на вбудовані пристрої [98].

7. Рівень управління та користувацьких застосунків забезпечує взаємодію КФС з операторами, інженерами, медичним персоналом чи іншими користувачами. На цьому рівні реалізуються інтерфейси моніторингу, візуалізації, налаштування політик доступу та сценаріїв автоматизації, інтеграція із зовнішніми інформаційними системами.

Користувацькі застосунки можуть бути реалізовані у вигляді SCADA-систем, веб-панелей адміністратора, мобільних додатків для віддаленого контролю, а також спеціалізованих інженерних консолей [99], [100]. Важливою складовою цього рівня є політики автентифікації та авторизації користувачів,

керування ролями й журналювання дій, що дозволяє відстежувати та аналізувати події, пов’язані з доступом до керуючих функцій та конфіденційних даних.

Узагальнення описаних рівнів дозволяє сформувати цілісну модель архітектури кіберфізичної системи, у якій кожен компонент взаємодіє з іншими через визначені функціональні канали та інформаційні потоки. На рисунку 1.1 подано семирівневу архітектуру КФС з відображенням взаємозв’язків між рівнями, напрямків руху даних та зон застосування криптографічних механізмів.

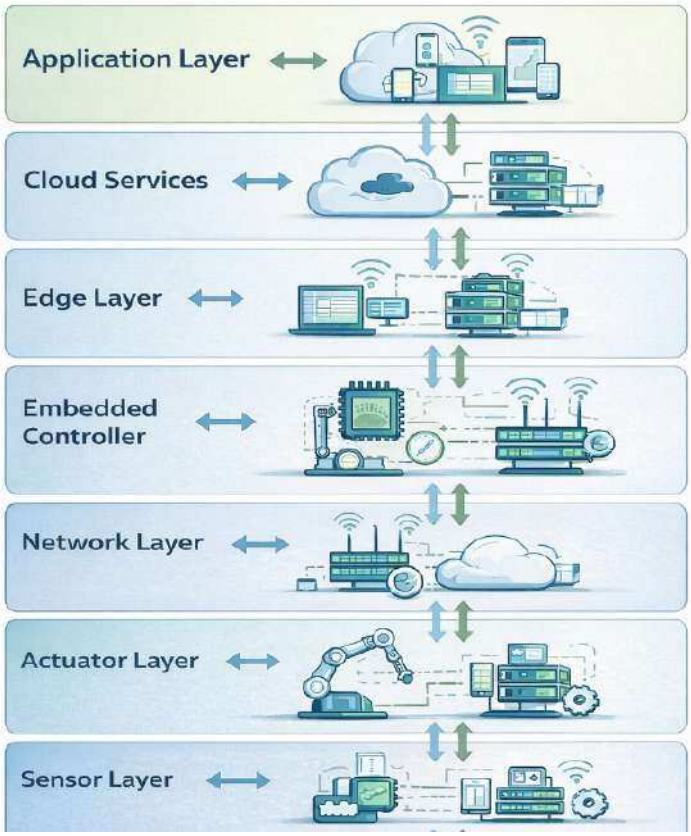

Рисунок 1.1 – Архітектура кіберфізичної системи

Архітектура відображає послідовну взаємодію фізичних, мережевих і сервісних шарів, що дає змогу визначити точки застосування криптографічних методів та сформувати вимоги до їх обчислювальної складності. На нижніх рівнях доцільним є використання полегшених механізмів захисту, орієнтованих на обмежені ресурси сенсорів і мікроконтролерів, тоді як у хмарних і прикладних рівнях можливе застосування повнофункціональних криптографічних алгоритмів. Така структура створює підґрунтя для побудови узгодженого

комплексу моделей і методів криптографічного захисту в компонентах КФС, що розробляються у подальших розділах дисертаційної роботи.

1.2. Класифікація загроз інформаційній безпеці у компонентах КФС

Компоненти КФС функціонують у тісному поєднанні фізичного середовища, програмного забезпечення та мережевих інтерфейсів. Така інтеграція робить їх вразливими до широкого спектра загроз, які можуть виникати як внаслідок фізичного впливу на пристрої, так і через атаки з віддалених джерел. Особливо складною є ситуація у випадках, коли пристрої мають обмежену обчислювальну потужність, малий обсяг пам’яті, нестабільне живлення або працюють без постійного адміністративного контролю.

Найчастіше загрози у КФС пов’язані з трьома аспектами: способом доступу до системи (фізичний або мережевий) [101], об’єктом атаки (програмне забезпечення, канали зв’язку, пам’ять) [102] і типом порушення основних властивостей інформації (цілісність, автентичність, конфіденційність) [103]. Деякі пристрої можуть бути виведені з ладу або скомпрометовані через фізичний доступ до мікроконтролера, зчитування вмісту Flash-пам’яті, втручання в інтерфейси налагодження або підміну вузлів у польових умовах. Інші ж стають жертвами віддалених атак, що використовують слабкі або застарілі протоколи зв’язку, відсутність автентифікації, віддалене оновлення без перевірки підпису тощо.

Критичними з точки зору інформаційної безпеки є також атаки на програмне забезпечення пристроїв [104]. Модифікація або підміна прошивки, ін’єктування зловмисного коду, використання недокументованих функцій, атак на механізми оновлення або відсутність захищеного завантаження (Secure Boot) можуть призвести до повного контролю над компонентом КФС [105]. Наслідком таких дій стає порушення роботи всієї системи, спотворення даних або перехоплення конфіденційної інформації.

Загальні типи загроз інформаційній безпеці у КФС систематизовано в таблиці 1.5, де наведено відповідність між каналами впливу, типом порушення та рівнем реалізації атаки в архітектурі.

Класифікація, представлена в таблиці, демонструє наскрізний характер загроз, що виникають у різних точках архітектури КФС. Водночас для ефективного протистояння цим загрозам необхідне поєднання кількох підходів: криптографічного захисту, перевірки цілісності, автентифікації пристроїв, аналізу поведінки та захисту на рівні завантаження. Проте в умовах обмежених ресурсів стандартні засоби безпеки не можуть бути застосовані без адаптації.

Таблиця 1.5 – Класифікація загроз інформаційній безпеці у компонентах КФС

Канал доступу	Об'єкт впливу	Тип порушення	Рівень архітектури КФС
Фізичний доступ	Пам’ять MCU, Flash, I/O	Порушення автентичності, витік ключів	Sensor, Actuator, Control Layer
Віддалений доступ	Мережевий інтерфейс, API	Порушення конфіденційності, перехоплення даних	Communication Layer
OTA-оновлення	Прошивка, Bootloader	Модифікація коду, підміна ПЗ	Embedded Control, Edge Layer
Нешифрований трафік	Транспортний рівень	Втрата цілісності, replay-атаки	Network, Edge, Cloud Layer
Скомпрометовані вузли	Імітація вузла в мережі	Порушення автентичності	Sensor, Actuator, Network
Вразливі протоколи	TLS/DTLS-lite, handshake	Підміна вузлів, MITM-атаки	Communication, Edge Layer

1.3. Аналіз сучасних підходів до криптографічного захисту інформації в умовах обмежених ресурсів

Криптографічний захист є одним із базових механізмів забезпечення інформаційної безпеки в кіберфізичних системах. У традиційних IT-середовищах використовуються потужні алгоритми з високим рівнем криптостійкості. Однак в умовах обмежених ресурсів, зокрема при використанні низькочастотних мікроконтролерів, обмеженого обсягу оперативної пам’яті, нестабільного живлення або батарейного функціонування, впровадження таких алгоритмів стає неприйнятним [106], [107]. Це зумовлює необхідність адаптації або розробки нових, полегшених криптографічних механізмів, які забезпечують баланс між безпекою, продуктивністю та енергоефективністю.

Серед найбільш активних напрямів розвитку у цій сфері виділяються полегшені симетричні та асиметричні алгоритми, адаптовані під ресурсообмежені середовища [108]. З огляду на архітектурні особливості мікроконтролерів, акцент робиться на мінімізацію кількості раундів шифрування, використання простих операцій (наприклад, XOR, побітових зсувів, підстановок) [109], зменшення потреби у пам’яті для ключів і таблиць підстановок [110], [111]. Однак при цьому важливо забезпечити криптографічну стійкість до відомих типів атак – диференціального, лінійного криптоаналізу, атак на основі часу виконання та споживаної енергії.

Ще одним ключовим напрямом є організація механізмів перевірки цілісності та автентичності програмного коду. З огляду на те, що компрометація прошивки може дати зловмиснику повний контроль над пристроєм, у сучасних підходах велика увага приділяється реалізації архітектури Secure Boot [112], хеш-контролю на рівні пам’яті мікроконтролера [113], використанню цифрових підписів, збережених у захищених сегментах. Адаптація таких рішень до малопотужних пристроїв потребує компромісів – наприклад, вибору коротших хешів або застосування симетричних підписів.

Окремий клас рішень зосереджується на захищеному оновленні програмного забезпечення, що є критичним для сучасних КФС, які експлуатуються тривалий час і потребують віддалених оновлень без фізичного доступу [114]–[117]. Основними проблемами тут є верифікація джерела оновлення, перевірка цілісності пакета, захист від атак типу «відкат до старої версії», а також керування сесіями та ключами під час оновлення в умовах нестабільного з'єднання або переривчастого живлення [118], [119].

Аналіз сучасних досліджень у цій галузі показує активне просування у створенні міжнародних стандартів, зокрема через ініціативу NIST Lightweight Cryptography [120], де були відібрані алгоритми-фіналісти з урахуванням вимог до продуктивності, пам’яті, енергоспоживання та безпеки.

1.3.1. Полегшені криптографічні алгоритми

Полегшені криптографічні алгоритми становлять окремий клас криптографічних примітивів, розроблених для середовищ з обмеженими ресурсами, зокрема сенсорних вузлів, мікроконтролерів та периферійних пристроїв кіберфізичних систем. Їх характерною особливістю є мінімізована складність операцій, низькі вимоги до обсягу оперативної пам’яті та коду, а також знижене енергоспоживання під час виконання криптографічних операцій [121], [122]. Такі властивості дозволяють забезпечувати криптографічний захист у системах, де застосування традиційних алгоритмів, наприклад AES або SHA-2, є технічно неможливим або призводить до невиправданих витрат ресурсів.

У відповідь на зростання потреби у стандартизованих рішеннях для IoT та систем з вбудованими пристроями Національний інститут стандартів і технологій США (NIST) у 2023–2025 роках завершив процес стандартизації полегшеної криптографії [123]. Результатом стало ухвалення сімейства алгоритмів Ascon як базового рішення для забезпечення шифрування з автентифікацією (AEAD), хешування та функцій з розширюваним виходом (XOF) [124]. Алгоритми Ascon

демонструють збалансоване співвідношення між криптостійкістю та ефективністю, мають компактний стан і використовують операції, оптимізовані під обмежені апаратні ресурси. Це робить їх придатними для реалізації в мікроконтролерах STM32Lx, STM32F4, ESP32-S3 та інших енергоефективних платформах [125], [126]. Поряд із Ascon, у сфері КФС поширення набули блокові шифри SPECK і SIMON, створені з орієнтацією на виконання у середовищах із жорсткими апаратними обмеженнями. Їх побудовано на основі ARX-операцій (додавання, циклічні зсуви, XOR) або спрощених побітових структур, що забезпечує високу швидкодію навіть у 16- або 32-бітних мікроконтролерах з мінімальним набором інструкцій [127], [128]. Ці алгоритми відзначаються низьким рівнем використання пам’яті та значно меншим енергоспоживанням у порівнянні з AES, однак їх криптостійкість залежить від конкретних конфігурацій параметрів, що потребує ретельного вибору режимів застосування.

У пристроях з обмеженими ресурсами, де обсяг доступної пам’яті вимірюється одиницями сотень байт, застосовуються полегшені криптопримітиви попередніх поколінь, такі як PRESENT, HIGHT, XTEA або інші полегшені конструкції [129], [130]. Їх архітектура базується на простих підстановках та перестановках, що знижує ресурсні вимоги, проте вони поступаються сучасним алгоритмам за рівнем стійкості до диференціального та лінійного криптоаналізу. Незважаючи на це, такі алгоритми залишаються актуальними у випадках, коли необхідно забезпечити базовий рівень захисту без значного навантаження на апаратну частину. У табл. 1.6 подано порівняльний огляд сучасних і класичних полегшених криптографічних алгоритмів, характерних для застосування у компонентах КФС, з урахуванням типових вимог до пам’яті, швидкодії та підтримуваних криптографічних функцій.

Таблиця 1.6 – Порівняльна характеристика полегшених криптографічних алгоритмів

Алгоритм	Тип крипто-примітиву	Основні операції	Обсяг коду (прибл.)	Пам’ять RAM	Підтримувані функції
Ascon (стандарт NIST)	AEAD, Hash, XOF	побітові операції, s-box, пермутації	2–4 КБ	< 1 КБ	шифрування + автентифікація, хешування
SPECK	блоковий шифр	ARX-операції	1–2 КБ	< 500 Б	шифрування
SIMON	блоковий шифр	прості побітові функції	1–2 КБ	< 500 Б	шифрування
PRESENT	блоковий шифр (SPN)	S-box, бітові перестановки	~1 КБ	< 200 Б	шифрування
HIGHT	блоковий шифр	побітові та арифметичні операції	~2 КБ	~200–400 Б	шифрування
XTEA	блоковий шифр	ARX-операції	<1 КБ	<200 Б	шифрування

Вибір конкретного криптографічного алгоритму для вбудованих компонентів кіберфізичних систем зумовлюється потребою у досягненні відповідного рівня стійкості до атак за умов обмежених ресурсів. Ураховуються не лише енергоефективність і швидкодія, а й сумісність із наявним апаратним забезпеченням, частота обміну даними та специфіка сценаріїв використання. Кожен алгоритм має власну придатність до певного класу пристроїв чи архітектур, що визначає його доцільність у тих чи інших умовах. Тому актуальні дослідження спрямовані на вдосконалення криптографічних рішень з урахуванням реальних обмежень і особливостей функціонування КФС, зокрема шляхом застосування полегшених конструкцій, що пройшли стандартизовані етапи тестування і рекомендовані органами стандартизації, зокрема NIST.

1.3.2. Методи перевірки цілісності та автентичності коду

Перевірка цілісності та автентичності програмного коду у вбудованих компонентах кіберфізичних систем має вирішальне значення для запобігання

запуску модифікованого, шкідливого або неавторизованого ПЗ. Особливо це актуально для пристроїв з обмеженими ресурсами, де неможливо реалізувати повноцінні захисні стеки, характерні для повнофункціональних комп'ютерів. До основних методів, що використовуються для контролю достовірності коду, належать захищене завантаження (Secure Boot), контроль хеш-сум і цифрові підписи [131], [132].

У сучасних кіберфізичних системах одним із ключових завдань є гарантування того, що на мікроконтролерах та інших вбудованих вузлах виконується тільки перевірений, авторизований програмний код, і будь-які спроби модифікації прошивки або підміни компонентів одразу виявляються. Для цього використовуються методи перевірки цілісності та автентичності коду, які реалізовано як апаратно-програмні механізми або програмні компоненти. Серед них – механізм secure boot, контроль хеш-сум, цифровий підпис прошивки, а також системи для безпечного оновлення.

Механізм Secure Boot надає фундамент для установлення довіри («root-of-trust») у пристрої: при кожному старті перевіряється, чи відповідає завантажуваний код очікуваним хешу або цифровому підпису, збереженому в апаратно-захищеній пам’яті [133]. У промислових платформах ця концепція реалізована виробниками. Наприклад, для серії мікроконтролерів STMicroelectronics існує пакунок X-CUBE-SBSFU (Secure Boot & Secure Firmware Update) [134], який дозволяє реалізувати завантаження лише перевіреного коду, шифрованих або підписаних прошивок та підтримує відкат у разі помилки під час оновлення.

Іншим прикладом є платформи на базі Espressif [135]: у серіях чипів ESP32 реалізовано secure boot з підтримкою RSA-PSS або ECDSA, де публічні ключі або їх дайджести зберігаються у захищених регістрах (eFuse), що унеможливлює зміну. При завантаженні перший-завантажувач (bootloader), другий-

завантажувач та образ застосунку перевіряються за підписом, що забезпечує ланцюг довіри – від ROM до кінцевого застосунку.

Таким чином, secure boot стає першочерговим засобом захисту прошивки, що обмежує виконання лише довіреного ПЗ і запобігає запуску модифікованих або шкідливих образів.

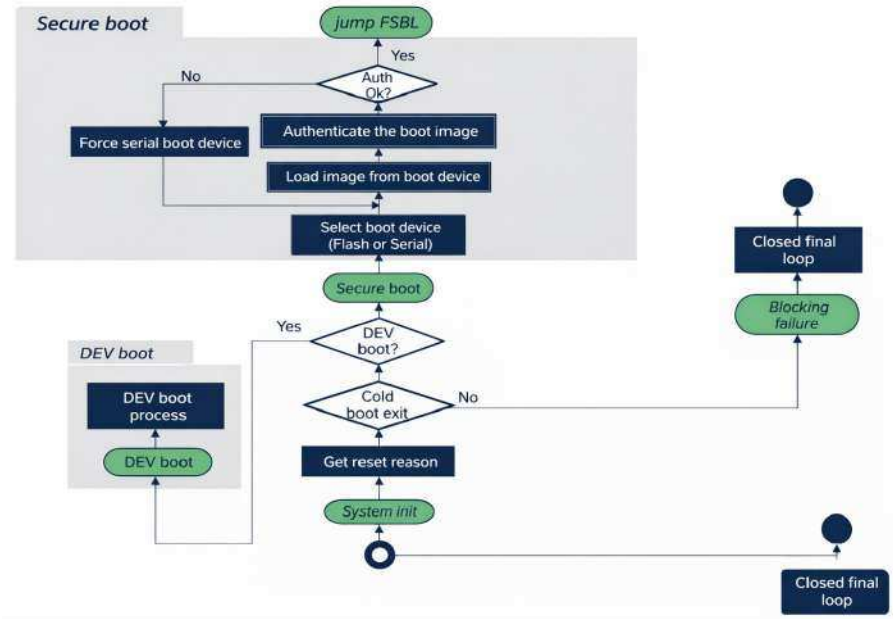

Рисунок 1.2 – Secure Boot у компонентах КФС

Після запуску з живлення (reset) первинний код (записаний у ROM або апаратно-захищеній зоні) перевіряє підпис або хеш наступного етапу завантаження. Кожна стадія bootloader→firmware→application має свою цифрову підпис або контрольну хеш-суму. Якщо перевірка проходить успішно – завантаження продовжується, інакше – блокування або перехід у режим відновлення. Така логіка забезпечує, що навіть у разі фізичного доступу або спроб заміни пам’яті, пристрій не запустить небажаний код.

Крім secure boot, широко використовують механізм контролю цілісності за допомогою хеш-функцій та цифрових підписів при оновленні прошивки або при завантаженні модулів під час роботи. Доступні opensource-та/або вендорські рішення – наприклад MCUboot або SWUpdate – підтримують підписані образи,

перевірку підписів RSA або ECDSA, контроль версій та rollback-захист [136], [137].

Такий підхід з цифровим підписом + хешування гарантує, що код походить із довіреного джерела, не був змінений, і відповідає очікуваній версії, що критично для систем з високим рівнем безпеки. У середовищах з обмеженими ресурсами часто використовують полегшені бібліотеки для ECC-підписів (наприклад, на базі ECDSA або EdDSA), що мають малий розмір коду і помірні вимоги до пам'яті, при цьому забезпечуючи прийнятний рівень безпеки [138], [139]. На рис. 1.3. представлено процес формування Контроль хеш-сум і цифрових підписів у вбудованих компонентах КФС.

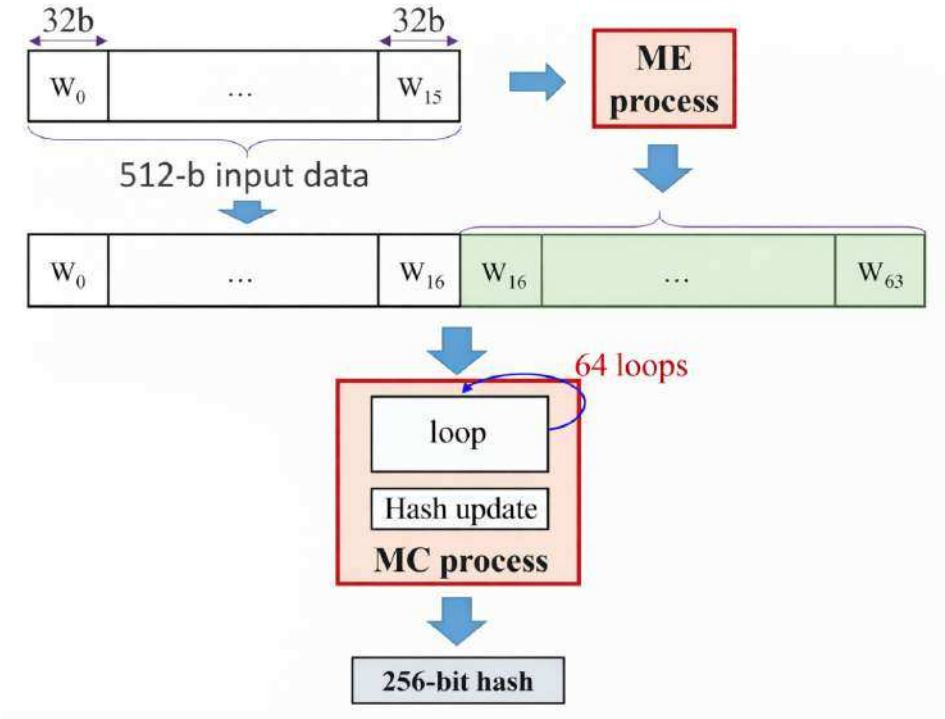

Рисунок 1.3 – Контроль хеш-сум і цифрових підписів у вбудованих компонентах КФС

На базовому рівні обчислюється хеш від образу прошивки (наприклад, SHA-256 або інша криптохеш-функція), який порівнюється з еталонним значенням, збереженим у захищеній зоні – при збігу дозволяється запуск, при невідповідності – відмова або перехід у режим відновлення [140]. При

використанні цифрової підпису перевірка відбувається шляхом криптографічної верифікації підпису за публічним ключем, що додає гарантію автентичності джерела. У випадках оновлення прошивки додаються контроль версії та механізми запобігання відкату (rollback), що унеможливлює повторне встановлення старих небезпечних версій.

Сучасні інструменти та фреймворки – secure boot, підписані прошивки, контроль версій, бібліотеки для ECC-підписів – дають практичну можливість реалізувати надійний захист коду у компонентах КФС навіть за умов обмежених ресурсів. Це робить їх ключовими елементами стратегії побудови безпечних кіберфізичних систем, адаптованих під обмеження апаратури.

1.3.3. Підходи до безпечного оновлення програмного забезпечення у КФС

У контексті кіберфізичних систем (КФС) з обмеженими обчислювальними ресурсами проблема безпечного оновлення програмного забезпечення (ПЗ) набуває особливої актуальності, адже будь-яке втручання у програмний стек пристрою може призвести до його виходу з ладу або компрометації критичних функцій. Забезпечення довіри до процесу оновлення передбачає використання верифікаційних механізмів, криптографічного захисту переданого коду, автентифікації джерела оновлення, а також ізольованих середовищ для виконання оновлень.

Захист від атак типу «програмне втручання» полягає в ізоляції оновлюваного середовища, перевірці цифрового підпису пакета оновлення до його застосування, а також забезпеченні неможливості зміни коду під час передачі. Для цього використовуються механізми автентифікації та шифрування, що працюють навіть на пристроях з обмеженою пам’яттю. Одним із дієвих підходів є реалізація фреймворку з двома розділами пам’яті – активним та резервним, де

оновлення спершу записується до резервного розділу, проходить перевірку, і лише потім здійснюється перемикання.

Верифікація джерела оновлення виконується шляхом перевірки підпису видавця оновлення. Пристрої мають зберігати попередньо вшитий відкритий ключ постачальника або кореневий сертифікат, на основі якого здійснюється перевірка. З метою економії пам’яті можуть використовуватись стислі форми сертифікатів, наприклад, X.509 з обмеженою кількістю полів [141]. У випадку використання широкомасштабного оновлення в сукупності пристроїв застосовуються моделі централізованого сервера оновлень, з TLS-з'єднанням та перевіркою сертифікатів обох сторін [142].

Управління ключами і сесіями при оновленнях реалізується із застосуванням криптографічних протоколів, таких як DTLS або OSCORE, або спрощених моделей з одноразовими ключами, що генеруються на основі часу або випадкових токенів [143], [144]. Одним із прикладів є використання тимчасових сесійних ключів, які дійсні лише на період оновлення та не зберігаються після завершення процесу. Це зменшує ризик довготривалого компрометування. Сучасні мікроконтролери, наприклад, лінійки STM32 або ESP32, мають вбудовану підтримку апаратних модулів зберігання ключів (Secure Element), які інтегруються у систему оновлення ПЗ та значно підвищують рівень стійкості до атак [145]–[147].

Підходи до безпечного оновлення в КФС також враховують обмеження каналів зв’язку, енергоспоживання та ризики ненадійного з’єднання. З цією метою застосовуються протоколи, що дозволяють відновлення процесу оновлення після збоїв, а також фрагментовану передачу прошивки з верифікацією кожного фрагмента. Одним з прикладів таких систем є фреймворки Mender [148], SWUpdate [149] або MCUboot [150], що адаптовані до мікроконтролерів і мають відкритий код для інтеграції у користувацькі прошивки. Завдяки можливості налаштування та вбудованим механізмам

безпеки, ці рішення дозволяють реалізувати гнучке та стійке до збоїв оновлення ПЗ, відповідаючи специфіці КФС.

1.4. Критерії та метрики ефективності криптографічного захисту в компонентах кіберфізичних систем

Оцінювання ефективності криптографічного захисту у компонентах КФС потребує комплексного підходу, який враховує технічні обмеження, специфіку функціонування в реальному часі, енергетичну автономність, а також вимоги до надійності й захищеності. На відміну від загальносистемного захисту в традиційних IT-інфраструктурах, у КФС кожен криптографічний механізм має бути адаптований до обмежень окремих підсистем – контролерів, сенсорів, вузлів збору даних та каналів передачі.

До основних критеріїв ефективності належать обчислювальні витрати, споживання енергії під час виконання криптографічних операцій, криптостійкість до базових і спеціалізованих типів атак, а також придатність до впровадження в реальні апаратні компоненти. При цьому важливим є забезпечення співвідношення між рівнем захисту та допустимим впливом на ресурси пристрою – зокрема, часом реакції системи, навантаженням на процесор і використанням енергоресурсів.

Систематизацію основних критеріїв та відповідних метрик оцінювання ефективності криптографічного захисту в компонентах кіберфізичних систем наведено в таблиці 1.7.

Для моделювання та тестування криптографічних функцій в КФС використано мову програмування C, орієнтовану на низькорівневий контроль, що дозволяє точно відстежувати обсяг використовуваної пам’яті, час виконання окремих інструкцій та звернення до периферійних ресурсів. Платформи на базі STM32, ESP32 та інші мікроконтролери з архітектурою ARM Cortex-M

забезпечують достатній рівень універсальності для експериментального аналізу криптографічного навантаження.

Таблиця 1.7 – Метрики ефективності криптографічного захисту в компонентах кіберфізичних систем

Критерій оцінювання	Основні метрики	Одиниці вимірювання	Метод оцінювання
Обчислювальна ефективність криптографічних механізмів	час виконання операції, кількість інструкцій, завантаження CPU	мс, кількість інструкцій, % CPU	профілювання виконання на мікроконтролері, аналіз інструкційного потоку
Енергоспоживання криптографічних операцій	енерговитрати на операцію, енерговитрати сеансу, частка енергії від батареї	мДж, Дж, %	вимірювання струму та напруги за допомогою Power Profiler, інтегральний розрахунок енергії
Криптостійкість та рівень безпеки	довжина ключа, ентропія ключового матеріалу, кількість раундів шифрування, стійкість до side-channel атак	біт, логарифмічна ентропія, кількість раундів	криптоаналітичний аналіз, відповідність стандартам NIST / ISO
Придатність до впровадження у компоненти КФС	обсяг ROM та RAM, залежність від апаратного прискорення, сумісність з архітектурою контролера	КБ, наявність / відсутність	експериментальна інтеграція алгоритмів у середовища STM32, ESP32, RISC-V

Обґрунтування вибору симуляційних і тестувальних інструментів ґрунтується на необхідності точного відтворення поведінки системи в реальному часі. Зокрема, використання FreeRTOS дозволяє відстежувати поведінку задач при одночасному виконанні кількох процесів криптографічної перевірки.

Інструменти типу Power Profiler Kit, Logic Analyzer, або STM32CubeMonitor дозволяють оцінити часові та енергетичні параметри кожної криптографічної операції [151], [152].

Підходи до верифікації включають проведення статичного та динамічного аналізу коду, емуляцію типових сценаріїв використання з фіксацією результатів, а також тестування на надійність при варіативному навантаженні, втраті зв’язку або збоях живлення. Інтеграція моделей загроз дозволяє оцінити потенційні вектори атак на кожному рівні – від стартової завантажувальної процедури до захисту передачі даних між модулями.

1.4.1. Обчислювальні витрати криптографічних механізмів

Оцінювання обчислювальних витрат криптографічних механізмів є ключовим етапом при впровадженні засобів захисту в компоненти кіберфізичних систем з обмеженими ресурсами. У контексті вбудованих пристроїв із низькою продуктивністю критично важливо забезпечити мінімальний час виконання криптографічних операцій, щоб не впливати на реальний час виконання основних функцій системи [153]. З цією метою проводиться аналіз затримки (latency) при виконанні операцій, що дозволяє виявити найпридатніші алгоритми для конкретного класу мікроконтролерів.

Важливим аспектом є навантаження на центральний процесор. Йдеться про кількість інструкцій, яку виконує криптографічна процедура, а також про відсоток часу, протягом якого обчислювальні ядра задіяні в обробці даних. Для реального часу критично, щоб криптографія не блокувала виконання інших задач, особливо на пристроях з одним ядром або без підтримки багатозадачності.

Особливу увагу приділено тривалості етапів генерації ключів, хешування, створення і перевірки цифрових підписів, які часто виконуються у фоновому режимі або під час встановлення з'єднання між пристроями. Для оцінювання затримок доцільно використовувати моделі, які враховують кількість інструкцій

N, тактову частоту контролера f, і середній час виконання однієї інструкції t_i. Узагальнено час обчислення можна подати як (1.1).

$$T_{exec} = N \cdot t_i = \frac{N}{f} \quad (1.1)$$

де T_{exec} – загальний час виконання операції. Ця модель дозволяє проводити формальне порівняння ефективності різних алгоритмів на однаковій апаратній платформі.

Математичне моделювання обчислювальних витрат зазвичай спирається на аналіз кількості елементарних операцій, що виконуються під час шифрування або дешифрування. Наприклад, загальний час виконання операції T_{op} можна подати як (1.2).

$$T_{op} = N_{instr} \cdot C_{clk} \cdot \frac{1}{f_{CPU}} \quad (1.2)$$

де N_{instr} – кількість інструкцій, необхідних для криптографічного алгоритму, C_{clk} – кількість тактів на інструкцію (CPI), f_{CPU} – тактова частота мікроконтролера. Залежно від архітектури, значення C_{clk} може варіюватися від 1 до 3, а в разі використання спеціалізованих інструкцій бути меншим за одиницю в середньому.

Особливу увагу слід приділяти часовим характеристикам ключових етапів криптографічної обробки: генерації ключів, обчислення хеш-функцій та перевірки цифрових підписів. Наприклад, для алгоритму ECC-256 генерація пари ключів на STM32F407 триває в середньому понад 1 секунду, що є неприйнятним для багатьох сценаріїв з низькою затримкою. Натомість алгоритми типу Ascon або TinyCrypt забезпечують хешування протягом десятків мілісекунд.

Зменшення кількості інструкцій, спрощення логіки S-box, уникнення багаторазового копіювання даних і використання апаратного прискорення дозволяє знизити витрати без шкоди для безпеки. Для емпіричної оцінки використано типові результати виконання базових криптографічних алгоритмів на архітектурі ARM Cortex-M. Зведені дані наведено в таблиці 1.8.

Таблиця 1.8 – Обчислювальні витрати полегшених криптографічних механізмів при реалізації на ARM Cortex-M

Алгоритм	Операція	Час виконання (мс)	Кількість інструкцій
Ascon-128	Хешування	15	~13 000
TinyCrypt (AES)	Шифрування блоків	27	~25 000
HIGHT	Шифрування	8	~9 000
PRESENT	Шифрування	10	~11 000
ECC-256	Генерація пари ключів	1230	~1 500 000
SPONGENT	Хешування	22	~18 000

Аналізуючи навантаження на обчислювальні ядра, доцільно враховувати рівень використання CPU під час криптографічних процедур. У випадку виконання шифрування паралельно з обробкою сенсорних даних, може виникнути конфлікт за ресурси, що спричинить зростання латентності в системі загалом. Зменшення кількості інструкцій, спрощення логіки S-box, уникнення багаторазового копіювання даних і використання апаратного прискорення дозволяє знизити витрати без шкоди для безпеки.

1.4.2. Енергоспоживання при виконанні криптографічних операцій

Оцінювання енергоспоживання криптографічних механізмів у компонентах кіберфізичних систем з обмеженими ресурсами є критичним аспектом, що впливає на тривалість автономної роботи пристроїв, зокрема сенсорних вузлів, контролерів та периферійних обчислювальних модулів [154]. У пристроях, живлення яких здійснюється від батареї або за рахунок енергозбирання, кожна криптографічна процедура повинна виконуватись із мінімальними витратами енергії без зниження рівня захищеності.

Оцінка енергоспоживання здійснюється з використанням інструментів типу Power Profiler Kit II, Otii Arc, Logic Analyzer, STM32CubeMonitor-Power, які дають змогу фіксувати миттєве та середнє значення струму споживання в процесі виконання криптографічних операцій [155], [156]. Отримані дані дозволяють визначити, які етапи шифрування, хешування або підпису є найбільш енерговитратними на конкретному мікроконтролері.

Загальне енергоспоживання E_{op} для операції визначається як (1.3).

$$E_{op} = I \cdot V \cdot T \tag{1.3}$$

де I – середній струм під час виконання операції (А), V – напруга живлення пристрою (В), T – час виконання криптографічної процедури (с).

У випадку використання динамічної зміни тактової частоти або переходу в режим сну після завершення обчислень, інтегральна оцінка енергоспоживання є точнішою за пікову. Для мікроконтролерів архітектури ARM Cortex-M, типове споживання струму становить від 4 до 15 мА у режимі активного виконання з тактовою частотою 48–168 МГц. Порівняльні результати для деяких поширених криптографічних алгоритмів наведено в таблиці 1.9.

Таблиця 1.9 – Енергоспоживання криптографічних алгоритмів на ARM Cortex-M (3.3 В, 72 МГц)

Алгоритм	Операція	Середній струм (мА)	Час виконання (мс)	Енерговитрати (мДж)
Ascon-128	Хешування	9,2	15	0,455
TinyCrypt (AES)	Шифрування блоків	10,1	27	0,898
PRESENT	Шифрування	8,5	10	0,281
ECC-256	Генерація пари ключів	12,3	1230	49,989
SPONGENT	Хешування	9,6	22	0,696

Оскільки енерговитрати прямо залежать від часу виконання, ключовим підходом до зменшення споживання є скорочення кількості інструкцій, оптимізація алгоритмів під апаратну реалізацію, використання прискорювачів шифрування або реалізація обчислень у енергоефективному режимі мікроконтролера. Крім того, у випадку повторюваних або періодичних криптографічних операцій, доцільним є застосування механізмів попередньої генерації ключів або буферизації результатів для уникнення зайвого дублювання обчислень.

На рівні системної інтеграції криптографія має мінімально впливати на графік переходів між енергетичними станами. Тривалі криптографічні процедури, які блокують вхід у режим сну, значно скорочують час автономної роботи, що неприпустимо для сенсорних вузлів з періодичним обміном повідомленнями. У зв’язку з цим оцінка енергоспоживання має проводитися не ізольовано, а в контексті циклу життєдіяльності пристрою, з урахуванням фаз пробудження, активної роботи та сну.

Сумарне енергоспоживання криптографічного сеансу $E_{session}$ визначається як сума енерговитрат усіх криптографічних процедур, що виконуються під час сеансу (1.4).

$$E_{session} = \sum_{i=1}^{n} E_i = \sum_{i=1}^{n} I_i \cdot V_i \cdot T_i \quad (1.4)$$

де n – кількість криптографічних операцій у межах одного сеансу (наприклад: шифрування, хешування, підпис, перевірка), I_i – середній струм споживання під час виконання і-ї операції, V_i – напруга живлення під час виконання і-ї операції (може бути сталою, наприклад 3.3 В), T_i – тривалість виконання і-ї операції.

У разі використання однакової напруги живлення для всіх операцій формула спрощується (1.5).

$$E_{session} = V \cdot \sum_{i=1}^{n} I_i \cdot T_i \quad (1.5)$$

Цей підхід дозволяє прогнозувати вплив на ресурс елементів живлення при варіативних сеансах автентифікації або підпису даних у медичних або сенсорних пристроях.

У пристроях з автономним живленням важливо враховувати кількість криптографічних сеансів на добу та середню тривалість сеансу. Для оцінки цього впливу розглядається модель (1.6).

$$E_{daily} = f \cdot E_{session} \tag{1.6}$$

де f – кількість криптографічних сеансів за одиницю часу (наприклад, за добу), $E_{session}$ – енергоспоживання одного сеансу, визначене вище.

Частка спожитої енергії відносно ємності акумулятора C_{bat} (у мДж) задається як (1.7).

$$P_{crypto} = \frac{E_{daily}}{C_{bat}} \cdot 100\% \tag{1.7}$$

Ця модель дозволяє кількісно оцінити, яку частку заряду акумулятора споживає криптографія щодня. Якщо значення $P_{crypto} > 20\%$, доцільно або зменшити частоту процедур, або адаптувати алгоритм, або реалізувати частину операцій у менш енерговитратному режимі мікроконтролера.

1.4.3. Оцінювання криптостійкості та рівня безпеки

Оцінювання криптостійкості криптографічних механізмів у компонентах кіберфізичних систем здійснюється за формальними параметрами захищеності, які визначають здатність алгоритмів протистояти базовим і спеціалізованим типам атак. Враховуються параметри, що характеризують захист від атаки повним перебором, побічного витоку (side-channel), повторного відтворення (replay), ін’єкції збоїв (fault injection) тощо.

Для забезпечення захисту від повного перебору ключа (brute force), довжина ключа повинна бути не меншою за 80 біт у пристроях з обмеженими ресурсами, тоді як для середнього рівня захищеності рекомендовані значення становлять 128

біт і вище. Криптостійкість залежить не лише від розміру ключа, але й від ентропії ключових матеріалів, яка характеризується логарифмічним показником невизначеності (1.8).

$$H(K) = -\sum_{i=1}^{n} p_i \cdot \log_2 p_i \quad (1.8)$$

де p_i – ймовірність появи і-го ключа. При рівномірному розподілі, що є оптимальним, ентропія $H(K)$ досягає максимуму, що збігається з довжиною ключа в бітах.

Циклічність шифрування – кількість раундів або циклів, необхідних для завершення криптографічної трансформації, – є критичним параметром у контексті забезпечення дифузії та стійкості до диференційного та лінійного криптоаналізу. Наприклад, більшість сучасних полегшених алгоритмів використовують від 10 до 20 раундів із повторенням перетворень, що ускладнює прогнозування внутрішнього стану навіть при частковому розкритті відкритого тексту [157].

Стійкість до побічних атак визначається через наявність або відсутність протоколів захисту від витоку інформації з каналів електромагнітного випромінювання, вимірювань споживаного струму або часу виконання. У вбудованих системах особливо актуальним є захист від Simple Power Analysis (SPA), Differential Power Analysis (DPA) та Fault Injection Attack (FIA), оскільки фізичний доступ до пристрою часто можливий [158], [159]. Захист передбачає випадкову перестановку інструкцій, введення фіктивних обчислень, використання маскування даних або дублювання обчислень із порівнянням результатів.

Для обґрунтованого порівняння алгоритмів за рівнем криптостійкості доцільно використовувати сукупність наступних метрик:

- довжина ключа (в бітах);
- рівень ентропії ключа;
- кількість раундів шифрування.

- наявність сертифікації згідно з NIST Lightweight Cryptography Finalist або ISO/IEC 29192;
- вразливість до side-channel атак (класифікація на основі експериментального профілювання);
- здатність до стійкої генерації цифрового підпису при вхідному шумі або збої.

Оцінювання відповідності стандартам реалізується через тестування сумісності з відкритими наборами векторів тестування (KATs – Known Answer Tests), які публікуються NIST, ISO/IEC або IETF [160]. Реалізація алгоритмів у середовищах з відкритим кодом (наприклад, TinyCrypt, mbedTLS, WolfSSL) [161] дає змогу інтегрувати верифіковані криптографічні ядра та скоротити кількість помилок, пов’язаних із власноручною реалізацією.

Для системного зіставлення параметрів криптостійкості полегшених алгоритмів шифрування в контексті КФС наведено порівняльні характеристики в таблиці 1.10.

Порівняльний аналіз алгоритмів, наведених у таблиці 1.10, свідчить про значну варіативність рівня криптостійкості залежно від архітектурних особливостей та реалізації контрзаходів. Алгоритми, розроблені в межах ініціативи NIST LWC, зокрема Ascon-128 та GIFT-128 [162], [163], демонструють підвищену стійкість до побічних атак за умов застосування маскування або інших засобів захисту. Разом з тим, класичні реалізації AES-128 без оптимізацій для вбудованих систем мають обмежену стійкість до side-channel атак, що знижує їхню доцільність у компонентах КФС без апаратної підтримки захисту. Стандартизованість і підтримка відповідними протоколами також впливають на вибір алгоритму для конкретного сценарію, особливо в контексті вимог до сертифікації безпеки.

Таблиця 1.10 – Криптостійкість сучасних полегшених алгоритмів шифрування

Алгоритм	Довжина ключа, біт	Кількість раундів	Стандартизація	Стійкість до side-channel атак
AES-128 (компактна реалізація)	28	10	NIST FIPS 197	Низька (без контрзаходів)
PRESENT	80/128	31	ISO/IEC 29192-2	Середня
GIFT-128	128	40	Finalist NIST LWC	Висока (за наявності маскування)
Ascon-128	128	12 (AEAD)/6 (hash)	NIST LWC Winner	Висока
TinyJAMBU	128	1024 стейтових кроків	Finalist NIST LWC	Середня
CLEFIA	128/192/256	18/22/26	Sony / ISO/IEC 29192-2	Середня
Speck-128/128	128	32	NSA (знято з розгляду)	Низька

1.4.4. Оцінювання придатності до впровадження у компоненти КФС

Впровадження криптографічних алгоритмів у апаратні компоненти КФС передбачає детальний аналіз їхньої сумісності з апаратними обмеженнями, підтримкою периферійних функцій, архітектурою контролерів, а також з вимогами до функціональності системи в реальному часі. Оскільки КФС охоплюють широкий спектр пристроїв – від енергонезалежних сенсорних вузлів

до потужніших керувальних контролерів – алгоритми шифрування мають демонструвати не лише високі параметри криптостійкості, але й реальну придатність до інтеграції з урахуванням апаратних і системних ресурсів.

Одним із ключових критеріїв є вимоги до пам’яті. Полегшені алгоритми повинні забезпечувати мінімальне використання як оперативної пам’яті (RAM), необхідної для зберігання проміжних результатів обчислень, так і постійної пам’яті (flash), де розміщується код алгоритму. Більшість типових пристроїв на базі ARM Cortex-M0+ або RISC-V в енергоефективному виконанні мають обмеження до 16–64 КБ flash і 2–8 КБ RAM [164]. У такому контексті вибір алгоритму з кодовою базою понад 20 КБ є недоцільним. Наприклад, реалізації Ascon-128 у варіанті AEAD споживають менш як 4 КБ ROM і 1 КБ RAM, що робить їх придатними до впровадження у сенсори або актуатори з обмеженими ресурсами [165].

Наступним критичним аспектом є сумісність з архітектурними особливостями мікроконтролерів. Алгоритми мають бути адаптовані до набору інструкцій конкретної архітектури – наприклад, використання ARM Thumb або RISC-V RV32I без розширень [166]. Наявність апаратного прискорювача AES або SHA може бути як перевагою, так і обмеженням: частина алгоритмів (наприклад, PRESENT, SPONGENT) не використовують ці ресурси, тому демонструють стабільну продуктивність на платформах без таких прискорювачів. Підтримка SIMD-інструкцій, багатоядерності або апаратного RNG також враховується при розробці реалізацій для КФС [167].

Оцінювання придатності також включає здатність алгоритму інтегруватися в типові сценарії використання. Зокрема, реалізація підтримки механізмів захищеного завантаження (Secure Boot), захищених оновлень прошивок (OTA), автентифікації в мережах IoT (наприклад, CoAP + DTLS), потребує алгоритмів із чітко визначеним інтерфейсом, можливістю інтеграції в стек протоколів і відповідністю стандартам. Застосування реалізацій з відкритим кодом (TinyCrypt,

Embedded Crypto Library, CryptoAuthLib) [168]–[170] полегшує впровадження й прискорює верифікацію.

Для системного порівняння криптографічних алгоритмів за критерієм придатності до впровадження доцільно враховувати такі параметри:

- обсяг ROM (flash) і RAM, необхідний для повноцінної реалізації;
- залежність від апаратного прискорення (чи потрібна підтримка AES engine, SHA unit);
- залежність від периферійних модулів (TRNG, таймери, DMA);
- наявність реалізацій для основних архітектур (ARM Cortex-M, RISC-V, MSP430);
- сумісність з протоколами захищеної комунікації;
- типові приклади використання у проєктах IoT, MedTech, Smart Grid, SCADA.

Таке багатопараметричне оцінювання дозволяє обґрунтувати вибір алгоритмів, які не лише відповідають вимогам до безпеки, але й можуть бути інтегровані в реальні компоненти КФС з урахуванням наявних апаратних обмежень і сценаріїв використання.

1.5. Постановка задач дисертаційного дослідження

Мета дисертаційного дослідження полягає у розробці методу та моделей, адаптованих до умов обмежених обчислювальних ресурсів, з урахуванням особливостей обміну даними та захищеного завантаження в архітектурах типу edge/fog.

У першому розділі дисертації виконано аналіз існуючих підходів до побудови систем захисту в умовах обмежених ресурсів на рівні вбудованих пристроїв та сенсорних модулів. Показано, що традиційні методи переважно орієнтовані на захист даних у централізованих системах і не враховують специфіку децентралізованої взаємодії між компонентами. Розглянуто механізми

захищеного завантаження, обміну інформацією та вибору криптографічних алгоритмів. Встановлено, що існуючі рішення або не забезпечують належного рівня безпеки, або є надмірно ресурсоємними для мікроконтролерів із низьким енергоспоживанням.

Аналіз методів захисту інформації під час завантаження вказує на доцільність впровадження архітектури Secure Boot з механізмами перевірки цілісності коду. Сформульовано задачу розробки удосконаленої моделі захищеного завантаження з урахуванням обмежень на обсяг пам’яті та продуктивність мікроконтролера.

Ще одним напрямком досліджень є побудова захищеного інформаційного обміну між компонентами в edge/fog-архітектурах. Аналіз існуючих протоколів встановлення захищених з’єднань показав, що більшість з них не оптимізовані для умов КФС, не враховують обмеження на затримку обміну та не забезпечують захист від атак повторної передачі. Сформульовано задачу побудови полегшених протоколів для встановлення захищених каналів з урахуванням динаміки маршрутизації та змінної топології.

Аналіз підходів до вибору криптографічних алгоритмів в умовах обмежених ресурсів виявив обмежену ефективність однофакторної оцінки. Показано необхідність використання багатокритеріального підходу, який враховує не лише енергоспоживання, а й продуктивність, криптостійкість та особливості реалізації на мікроконтролерах різних класів. Сформульовано задачу розробки методу обґрунтованого вибору криптографічних алгоритмів.

Окрім того, встановлено відсутність інтегрованих моделей, які поєднують захищене завантаження, вибір криптографічного алгоритму та захищений обмін у межах одного підходу. Сформульовано задачу побудови комплексної системи криптографічного захисту інформації, орієнтованої на інтеграцію всіх компонентів у КФС.

Виходячи з цього, задачами роботи є:

1) удосконалити модель захисту інформації на рівні вбудованих пристроїв та сенсорних модулів**,** яка включає механізми захищеного завантаження, контролю цілісності програмного коду, вибору та інтеграції криптографічних алгоритмів, а також управління ключовим матеріалом з урахуванням обмежених обчислювальних ресурсів;
2) розробити модель криптографічного захисту інформаційного обміну між компонентами кіберфізичних систем**,** яка враховує динамічну топологію, особливості децентралізованих архітектур типу edge/fog та забезпечує протидію атакам на канали передавання, зокрема атакам повторної передачі, підміни та перехоплення даних;
3) удосконалити модель динамічного розподілу криптографічного навантаження між компонентами кіберфізичної системи та периферійними обчисленнями**,** яка базується на врахуванні стану ресурсів, характеристик каналів зв’язку та вимог до безпеки і забезпечує адаптивний вибір локального або делегованого виконання криптографічних операцій;
4) розвинути метод побудови комплексної системи криптографічного захисту інформації у компонентах кіберфізичних систем**,** який інтегрує механізми захищеного завантаження, вибору криптографічних алгоритмів, організації захищеного обміну та розподілу криптографічного навантаження на основі визначених критеріїв і правил;
5) оцінити ефективність запропонованого методу та моделей у програмно-апаратному середовищі з використанням мікроконтролера STM32 за визначеними метриками (обчислювальні витрати, енергоспоживання, затримки, продуктивність) та здійснити порівняння з існуючими криптографічними рішеннями.

Поставлені задачі дисертаційного дослідження несуть наукову новизну і практичну цінність, а їх вирішення дозволить досягти поставленої мети.

1.6. Висновки до розділу 1

У першому розділі проведено комплексне дослідження сучасного стану предметної області, пов’язаної із захистом інформації в компонентах кіберфізичних систем за умов обмежених обчислювальних ресурсів. Детально проаналізовано архітектурну структуру КФС, визначено ключові рівні взаємодії та функціональні характеристики сенсорних вузлів, мікроконтролерів, периферійних обчислювальних середовищ і хмарних сервісів, що дозволило чітко окреслити зони виникнення потенційних загроз і точки застосування криптографічних механізмів.

Виконано розширену класифікацію типових загроз, які охоплюють фізичний, мережевий і програмний рівні доступу до компонентів КФС, а також визначено відповідні рівні архітектури, на яких ці загрози реалізуються. Окрему увагу приділено атакам, спрямованим на процеси завантаження та оновлення програмного забезпечення, підміну вузлів у мережевому середовищі, компрометацію пам’яті пристроїв і порушення автентичності даних.

Проведено поглиблений аналіз сучасних підходів до криптографічного захисту в умовах ресурсних обмежень, зокрема полегшених симетричних та асиметричних алгоритмів, методів контролю цілісності програмного коду, механізмів Secure Boot, цифрових підписів, а також сучасних підходів до організації безпечного оновлення програмного забезпечення у вбудованих системах.

Виконано порівняльний аналіз сучасних полегшених криптографічних алгоритмів (Ascon, PRESENT, GIFT-128, TinyCrypt тощо). Оцінювання проведено з позиції обчислювальних витрат, енергоспоживання, криптостійкості та придатності до впровадження у мікроконтролерах з обмеженими ресурсами. На цій основі побудовано аналітичні моделі для оцінки часу виконання криптографічних операцій, енерговитрат, а також впливу криптографічних процедур на ресурс акумуляторного живлення пристрою.

Сформовано узагальнені критерії та метрики ефективності криптографічного захисту, що включають затримку обробки, рівень завантаження процесора, споживання енергії, довжину ключа, кількість раундів, рівень ентропії, стійкість до побічних атак і відповідність сучасним стандартам безпеки. Запропоновано методику багатокритеріального аналізу криптографічних алгоритмів із урахуванням архітектурних і функціональних особливостей КФС.

У підрозділі 1.5 сформульовано мету дослідження, об’єкт і предмет, а також задачі дисертаційної роботи, що включають удосконалення моделі захищеного завантаження, розробку методу вибору криптографічного алгоритму, побудову моделі захищеного обміну в edge/fog-архітектурах, визначення метрик ефективності, створення методології комплексного захисту та експериментальну перевірку результатів на мікроконтролері STM32.

2. МОДЕЛЬ ЗАХИСТУ ІНФОРМАЦІЇ НА РІВНІ ВБУДОВАНИХ ПРИСТРОЇВ ТА СЕНСОРНИХ МОДУЛІВ

2.1. Структура сенсорного рівня кіберфізичних систем та визначення вимог до інформаційної безпеки

У КФС сенсорні компоненти забезпечують первинне зчитування фізичних величин із середовища та їх перетворення в цифрову форму для подальшої обробки. Архітектура рівня сенсорних компонентів передбачає використання вузлів збору даних, які, як правило, реалізовані на базі мікроконтролерів з обмеженими обчислювальними можливостями, обсягом пам’яті та автономним енергоживленням. Основними функціональними блоками таких вузлів є датчики, аналого-цифрові перетворювачі, модулі попередньої обробки, модулі передачі даних (наприклад, через UART, SPI, I²C або радіоканали типу BLE, ZigBee), блоки енергозабезпечення та енергоефективного керування. Комунікація між компонентами часто здійснюється з використанням відкритих протоколів, що підвищує ризик атак на рівні передавання та зберігання даних.

На рисунку 2.1 представлено структурну архітектуру сенсорного рівня у межах вбудованої платформи КФС. Виділено основні функціональні блоки та канали даних, які виступають потенційними точками входу для атак.

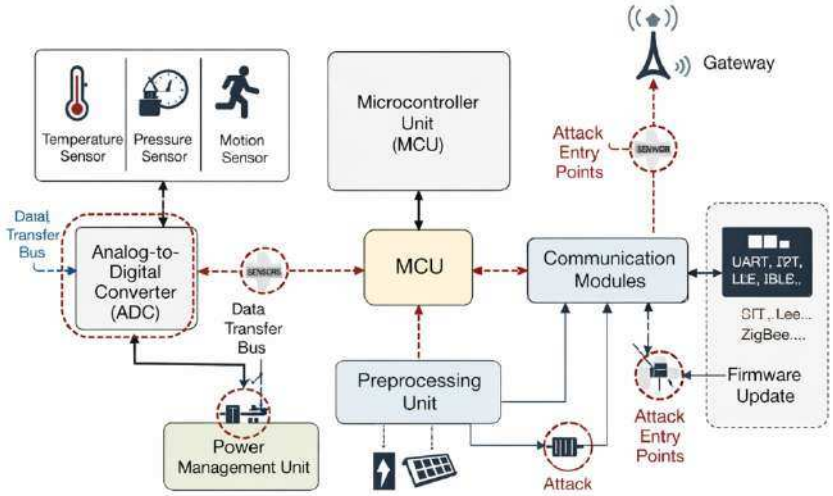

Рисунок 2.1 – Структурна схема архітектури рівня сенсорних компонентів КФС

Вразливість сенсорних вузлів зумовлена як їх розташуванням у відкритому фізичному середовищі, так і обмеженими ресурсами, що унеможливлює застосування традиційних криптографічних протоколів повного спектру. До обмежень належать: обмежений обсяг оперативної пам’яті (у межах 2–256 КБ), обмежена частота процесора (до 200 МГц), низький рівень енергоспоживання (до 50 мВт), відсутність апаратного прискорення криптографічних операцій, обмеження щодо розміру прошивки.

У такій архітектурі потенційними точками входу для атак є: шина передачі даних між датчиком і мікроконтролером, внутрішня пам’ять, де тимчасово зберігаються необроблені або оброблені дані, канал зв’язку між сенсором і шлюзом (gateway), а також інтерфейс для оновлення прошивки. Типові атаки на цьому рівні включають перехоплення даних (eavesdropping), підміну (tampering), ін’єкцію даних (data injection), перехоплення або підміну прошивки, атакування через побічні канали (наприклад, аналіз споживання енергії або електромагнітного випромінювання), а також фізичний доступ до пристрою з метою копіювання або підміни ключів.

2.2. Модель захищеного завантаження програмного коду сенсорного вузла

Процес завантаження програмного забезпечення в сенсорні вузли кіберфізичних систем є критичним етапом, під час якого виконується ініціалізація всіх функціональних блоків пристрою. На цьому етапі забезпечення автентичності та цілісності програмного коду є необхідною умовою для запобігання навмисній модифікації або підміні прошивки з боку зловмисника.

У більшості сучасних вбудованих платформ початковий етап виконання програмного коду організовано за принципом Secure Boot, який передбачає перевірку автентичності та цілісності програмного забезпечення перед його запуском [76]. Такий підхід дозволяє сформувати ланцюг довіри, у якому кожний

наступний компонент системи запускається лише після перевірки попереднім компонентом.

Модель захищеного завантаження розглядається не лише як алгоритм перевірки прошивки, а як комплексна архітектурна схема взаємодії механізмів контролю цілісності, перевірки цифрового підпису, захищеного зберігання ключового матеріалу та контролю версій програмного забезпечення [105]. Такий підхід враховує специфіку сенсорних вузлів кіберфізичних систем, для яких характерні обмежені обчислювальні ресурси, невеликий обсяг пам'яті та жорсткі вимоги до енергоспоживання.

Загальну структуру запропонованої моделі захищеного завантаження сенсорного вузла наведено на рисунку 2.2.

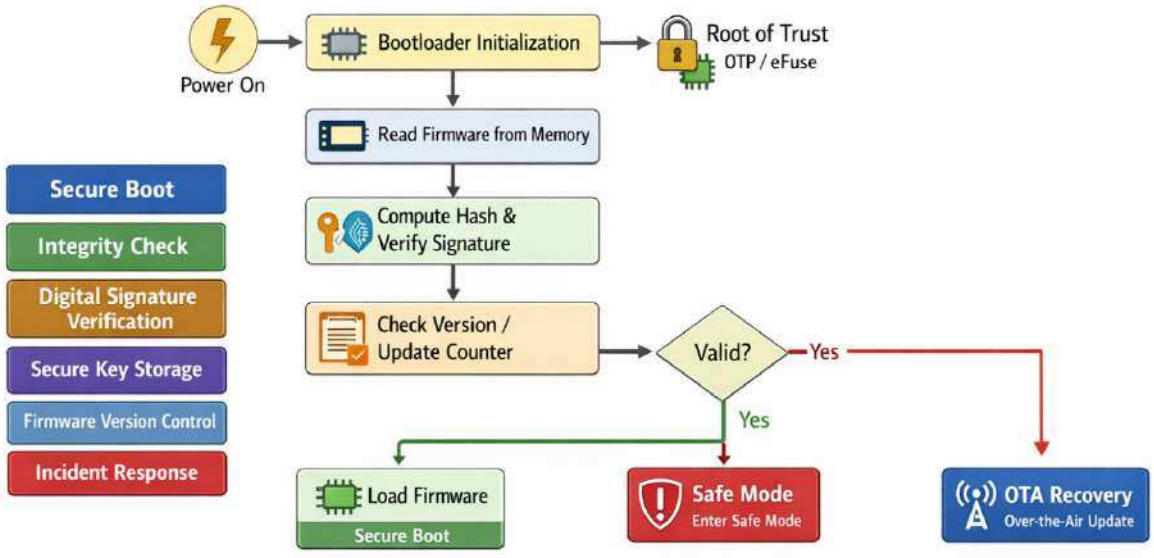

Рисунок 2.2. – Структурна схема архітектури захищеного завантаження сенсорного вузла КФС

Представлена модель включає декілька взаємопов'язаних компонентів. Після подачі живлення або перезавантаження пристрою ініціалізація починається з виконання початкового коду bootloader, який розміщується у захищеній області пам'яті мікроконтролера. Bootloader використовує корінь довіри, що зберігається

у незмінній пам’яті типу OTP або eFuse, для перевірки автентичності програмного забезпечення.

На наступному етапі виконується зчитування образу прошивки з пам’яті пристрою або з каналу оновлення. Після цього обчислюється криптографічне хеш-значення програмного коду та виконується перевірка цифрового підпису. У разі відповідності підпису збереженому публічному ключу та підтвердження цілісності програмного коду система переходить до перевірки версії прошивки. Контроль версії дозволяє запобігти downgrade-атакам, у яких зловмисник намагається завантажити стару версію програмного забезпечення з відомими вразливостями.

Якщо всі перевірки завершуються успішно, відбувається запуск основної прошивки пристрою. У випадку виявлення невідповідності або порушення цілісності програмного коду система переходить у безпечний режим або ініціює процедуру відновлення прошивки через захищений канал оновлення.

Представлена структурна схема визначає основні функціональні компоненти моделі захищеного завантаження. Для реалізації цієї архітектури використовується послідовний алгоритм перевірки програмного забезпечення під час запуску пристрою. Такий алгоритм базується на роботі bootloader та передбачає виконання криптографічної перевірки програмного коду перед передачею керування основній прошивці. Послідовність перевірки програмного коду під час захищеного завантаження показано на рисунку 2.3.

На рисунку 2.3 представлено загальну послідовність дій у процесі захищеного завантаження програмного забезпечення із перевіркою цілісності, яка реалізується в умовах обмежених ресурсів вбудованих пристроїв.

Алгоритм функціонування типового захищеного завантаження включає наступні етапи:

1. Запуск bootloader після подачі живлення.

2. Зчитування основної прошивки з пам’яті або з зовнішнього джерела (наприклад, OTA-оновлення).
3. Обчислення контрольної хеш-сумми або перевірка цифрового підпису на основі вбудованого публічного ключа.
4. Порівняння результату з очікуваним еталонним значенням.
5. Рішення про запуск основної програми або перехід у безпечний режим у разі невідповідності.

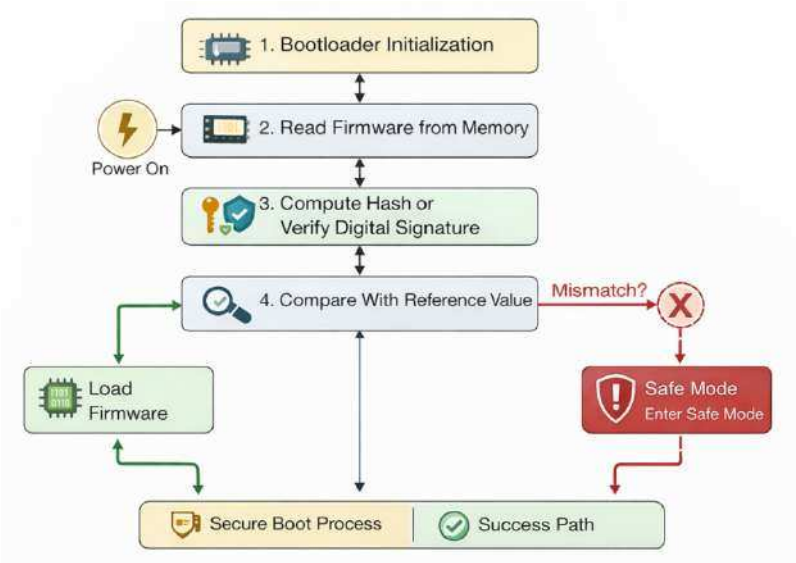

Рисунок 2.3 – Модель захищеного завантаження із перевіркою цілісності коду

Цілісність коду в більшості реалізацій перевіряється за допомогою алгоритмів хешування (SHA-256, BLAKE2s тощо), а автентичність – за допомогою електронного цифрового підпису (ECDSA, Ed25519). У разі використання лише хеш-сум, обов’язковим є захист еталонного значення, яке зберігається у bootloader або у заздалегідь відомому надійному сховищі.

У реальних реалізаціях, зокрема на платформах STM32 процес захищеного завантаження може бути частково реалізований на апаратному рівні (наприклад, з використанням OTP-пам’яті для збереження публічного ключа та механізмів Secure Boot, підтримуваних виробником мікроконтролера).

З точки зору атаки, відсутність контролю цілісності на етапі завантаження дозволяє зловмиснику виконати так звану bootloader replacement attack, коли на

місце легітимного програмного забезпечення завантажується модифікований код із прихованою шкідливою функціональністю. Крім того, можливі атаки на сторонні канали – наприклад, через аналіз енергоспоживання при верифікації коду або атаки повторного використання прошивок з іншими ключами.

2.2.1. Архітектура Secure Boot у вбудованих пристроях

Secure Boot у вбудованих системах – це механізм захисту, який забезпечує криптографічну перевірку автентичності та цілісності програмного забезпечення до його виконання [105]. Його основна мета – запобігання запуску модифікованої або неавторизованої прошивки на мікроконтролері. Механізм базується на концепції «довіреного кореня» (root of trust), коли початковий завантажувач (bootloader) є єдиним джерелом, якому система повністю довіряє, а всі наступні компоненти верифікуються криптографічно перед передаванням їм управління. Його основне призначення – гарантувати, що лише довірене та не модифіковане програмне забезпечення може бути виконане на пристрої. Це досягається за рахунок криптографічної перевірки автентичності та цілісності завантажуваного коду перед його запуском.

Secure Boot реалізується на стику апаратного забезпечення та програмного середовища. Зокрема, більшість сучасних мікроконтролерів підтримують або апаратне розширення (наприклад, OTP-пам’ять, апаратні модулі хешування), або фреймворки для реалізації програмної частини Secure Boot.

На рисунку 2.4 показано архітектуру Secure Boot, яка включає основні елементи: захищене сховище кореневого публічного ключа, bootloader з механізмами верифікації, захищену область збереження прошивки, блок контролю цілісності та систему реагування на невідповідність.

Основні компоненти архітектури Secure Boot:

1. Кореневий публічний ключ Root of Trust (RoT). Зберігається у забороненій до модифікації області (наприклад, у OTP або eFuse) та використовується для перевірки цифрового підпису завантажуваного коду.

2. Bootloader з криптографічною верифікацією. Це перший виконуваний код. Його завдання – зчитати прошивку, обчислити її хеш-суму та перевірити цифровий підпис. У разі успішної верифікації управління передається основному додатку. У разі помилки – ініціюється блокування, перезавантаження або активація безпечного режиму.

3. Пам’ять з прошивкою (Application image). Прошивка зберігається у зовнішній або внутрішній флеш-пам’яті та супроводжується метаданими: хеш-сумою та цифровим підписом.

4. Блок верифікації коду. Реалізується за допомогою вбудованих криптографічних бібліотек або апаратних прискорювачів (наприклад, SHA-256 engine або RSA/ECC engine).

5. Механізм реакції на помилку. При виявленні невідповідності (змінений хеш або недійсний підпис) завантаження основної програми не відбувається. Пристрій залишається в захищеному стані.

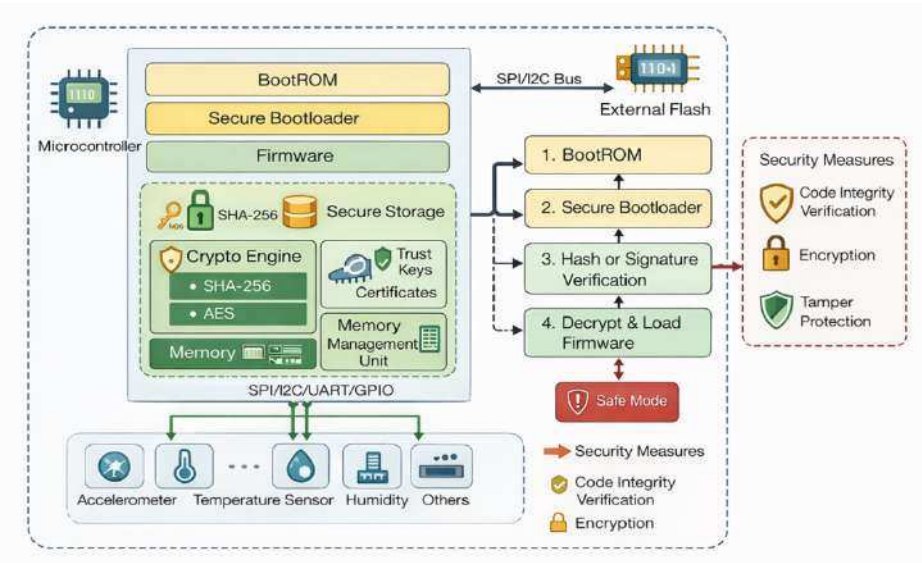

Рисунок 2.4 – Архітектура Secure Boot для сенсорного пристрою на базі мікроконтролера

Механізм перевірки цифрового підпису реалізується у вигляді програмного модуля в середовищі завантажувача. Його завдання – обчислити хеш контрольної суми прошивки та перевірити автентичність отриманих даних шляхом верифікації електронного підпису. Реалізація цього процесу для сенсорного пристрою на базі мікроконтролера STM32 представлена у вигляді фрагмента програмного коду, наведеного у Додатку А.1.

Реалізація алгоритму протестована на апаратних платформах STM32F103 та ESP32-S3 – популярних представниках 32-бітних мікроконтролерів для вбудованих систем. Завантаження прошивки відбувається з зовнішньої SPI Flash або в процесі оновлення «по повітрю» (OTA, Over-the-Air). Під час експериментального дослідження здійснено вимірювання часу, енергоспоживання, використання пам’яті та обсягу коду, необхідного для реалізації повного Secure Boot-процесу.

Поетапна реалізація перевірки цілісності та автентичності прошивки у вбудованих системах із обмеженими ресурсами включає низку взаємопов’язаних дій, які реалізуються в рамках механізму Secure Boot. Цей механізм забезпечує криптографічний контроль легітимності програмного забезпечення на етапі завантаження, мінімізуючи ризики запуску модифікованого або шкідливого коду. Послідовність дій у межах реалізації Secure Boot на мікроконтролерах STM32F103 або ESP32-S3 під керуванням FreeRTOS чи у bare-metal конфігурації може бути описана наступним чином:

1. Після подачі живлення або перезавантаження мікроконтролера виконується ініціалізація bootloader, розміщеного в захищеній області пам’яті. На цьому етапі налаштовується тактування, периферія та інтерфейси (SPI, USART) для доступу до зовнішньої Flash-пам’яті, а також ініціалізуються криптографічні бібліотеки (наприклад, mbedTLS або TweetNaCl).

2. Bootloader зчитує метадані прошивки: розмір, контрольну суму та цифровий підпис. Дані завантажуються блоками, що дозволяє зменшити навантаження на оперативну пам’ять під час обробки.

3. Обчислюється хеш від основного тіла прошивки. Для STM32F103 рекомендовано використовувати BLAKE2s, яка забезпечує баланс між продуктивністю та енергоспоживанням. У ESP32-S3 може бути використано SHA-256 завдяки апаратній підтримці.

4. Виконується перевірка цифрового підпису шляхом порівняння підписаного еталонного хешу з обчисленим. Публічний ключ зберігається у захищеній пам’яті (наприклад, flash або eFuse). Підтримуються алгоритми ECDSA або Ed25519.

5. У разі успішної перевірки bootloader передає керування основній програмі, змінюючи адресу таблиці переривань. Якщо перевірка не пройдена, пристрій переходить у захищений режим (наприклад, очікування оновлення або активація резервної версії прошивки).

6. На етапі тестування запропонована реалізація продемонструвала високу ефективність. Зокрема, на STM32F103 хешування 1 КБ даних за допомогою BLAKE2s займає 2.3 мс, перевірка підпису ECDSA – 6.5 мс, що у сумі дає приблизно 12 мс Secure Boot. На ESP32-S3 з SHA-256 та Ed25519 ці значення становлять відповідно 4.1 мс та 1.1 мс, що дозволяє завершити перевірку за близько 6 мс.

Запропонована схема доводить можливість безпечного запуску програмного забезпечення навіть на пристроях з обмеженими обчислювальними ресурсами, зберігаючи при цьому низьке енергоспоживання та високу надійність криптографічного контролю.

2.2.2. Методи хешування та цифрового підпису для перевірки автентичності коду

Перевірка цілісності та автентичності програмного коду у вбудованих пристроях ґрунтується на комбінації алгоритмів хешування та механізмів електронного цифрового підпису [171]. У контексті обмежених ресурсів сенсорних вузлів, важливо обирати криптографічні методи з низьким навантаженням на процесор і мінімальним енергоспоживанням.

На рисунку 2.5 представлено загальну схему процесу перевірки цифрового підпису в межах завантаження прошивки. Цей процес поділяється на три логічні етапи: хешування вхідного блоку даних, перевірка автентичності через цифровий підпис та прийняття рішення щодо подальшого виконання коду.

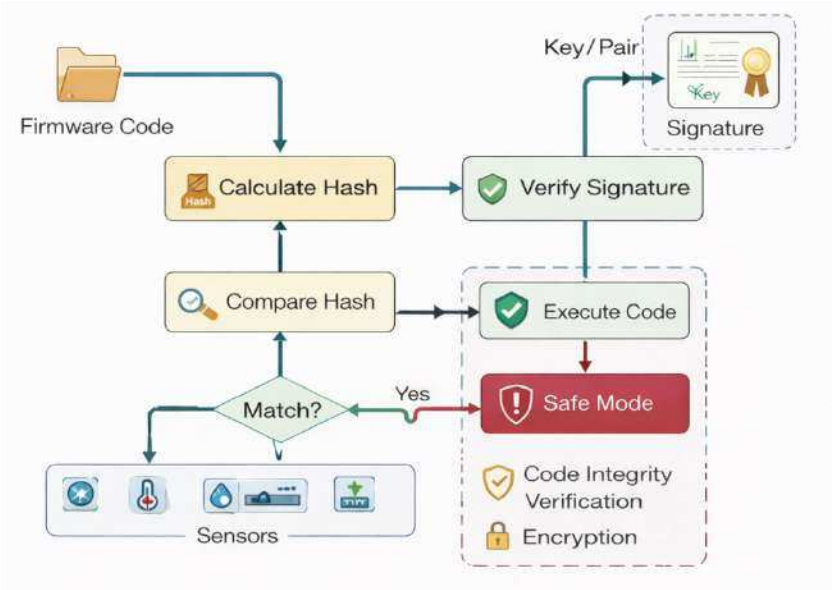

Рисунок 2.5 – Схема перевірки цілісності та автентичності програмного коду

Згідно зі схемою, наведеною на рисунку 2.5, процес перевірки автентичності полягає у послідовному обчисленні хеш-функції від отриманого програмного коду, порівнянні її з контрольним значенням та перевірці цифрового підпису за допомогою відкритого ключа. У випадку успішного проходження верифікації, прошивка вважається автентичною, а її виконання дозволяється [172]. У разі виявлення невідповідності або пошкодження підпису, ініціюється перехід у

захищений режим, що унеможливлює подальше використання потенційно скомпрометованого коду. Такий підхід гарантує, що вбудований пристрій працює лише з перевіреним програмним забезпеченням, джерело походження якого підтверджено криптографічно.

У загальному випадку процес автентифікації прошивки можна подати як операцію порівняння обчисленого хеш-значення з референтним підписаним хешем, що визначає валідність програмного коду та описується математичною моделлю (2.1).

$$Verify(P, \sigma, PK) = \begin{cases} true, \text{якщо } Sign^{-1}(\sigma, PK) = H(P) \\ false, \text{інакше} \end{cases} \quad (2.1)$$

де P – вміст прошивки (програмного коду), що перевіряється; σ – електронний цифровий підпис, отриманий на етапі генерації прошивки; PK – публічний ключ, що зберігається в bootloader або в незмінюваному сховищі мікроконтролера; $H(P)$ – результат хешування прошивки за допомогою криптографічного хеш-функціоналу (наприклад, SHA-256); $Sign^{-1}$ – результат верифікації підпису, який має збігатися з хеш-значенням прошивки $Verify$ – булева функція, що повертає істину (true) у разі успішної автентифікації, або хибність (false) при невідповідності.

Ця модель відображає базову логіку цифрової верифікації у Secure Boot-сценарії, де довіра базується на криптографічному корені довіри, збереженому в пристрої.

У вбудованих сенсорних пристроях, що функціонують в умовах обмежених обчислювальних ресурсів, вибір хеш-функції має ґрунтуватися на компромісі між безпекою, швидкодією, енергоспоживанням і вимогами до обсягу пам’яті. Класичні хеш-функції, що широко застосовуються у повнофункціональних обчислювальних системах, не завжди придатні для реалізації в мікроконтролерах з обмеженим обсягом ОЗП, ПЗП або з автономним живленням. Тому пріоритет надається хеш-функціям, які мають компактну реалізацію, допускають часткову

апаратну підтримку й забезпечують достатній рівень стійкості до криптографічних атак.

До таких функцій належить BLAKE2s, яка вважається однією з найшвидших серед сучасних хеш-функцій у категорії програм із відкритим кодом, адаптованих для 8-, 16- та 32-бітних мікроконтролерів [173]. Вона демонструє високу продуктивність і енергоефективність навіть за умов відсутності апаратного прискорювача. SHA-256, попри свою поширеність і підтримку стандартами безпеки, споживає більше енергії та має вищі вимоги до обчислювальних ресурсів, однак її підтримка в апаратному модулі деяких мікроконтролерів (зокрема STM32) компенсує ці недоліки. Інноваційним підходом є використання KangarooTwelve (K12) [174] – полегшеної модифікації Keccak, яка дозволяє реалізувати потокове хешування й адаптується під довгі повідомлення, при цьому залишаючись компактною у реалізації. Для підтвердження ефективності вибраних хеш-функцій було проведено експериментальне тестування на трьох платформах, які широко використовуються у вбудованих системах:

- STM32F103C8T6 (архітектура ARM Cortex-M3, тактова частота 72 МГц, 20 КБ RAM, 64 КБ Flash) – як представник класичних 32-бітних мікроконтролерів із базовими ресурсами;
- ESP32-S3 (двоядерна архітектура Xtensa LX7, до 240 МГц, 512 КБ SRAM) – сучасний варіант з більшою обчислювальною потужністю та апаратною підтримкою криптоалгоритмів;
- Longan Nano на базі GD32VF103 (архітектура RISC-V, 108 МГц, 32 КБ SRAM) – для оцінки продуктивності на відкритій архітектурі з перспективою використання у бюджетних сенсорних модулях.

Вимірювались час виконання, обсяг спожитої оперативної пам’яті (RAM), розмір коду (Flash) і енерговитрати під час обчислення хешу. Підсумкові значення зведені в таблицю 2.1. Продуктивність означає обчислювальну

швидкість хеш-функції на цільовому мікроконтролері, тобто наскільки швидко вона обробляє дані за заданих апаратних обмежень.

Таблиця 2.1 – Порівняння продуктивності та енергоефективності хеш-функцій у сенсорних мікроконтролерах

Хеш-функція	Кількість тактів (на 1 KB)	Пам’ять (RAM)	Розмір коду (Flash)	Функціональні особливості
SHA-256	3 800	~1.2 KB	~5.5 KB	Стандартна стійкість
BLAKE2s	1 300	~1.0 KB	~3.5 KB	Висока швидкість
K12	2 100	~0.8 KB	~4.0 KB	Гнучка структура

У контексті моделі захисту інформації на рівні вбудованих пристроїв та сенсорних модулів особливу увагу приділено механізму перевірки цілісності програмного коду. Саме цей механізм забезпечує першу лінію захисту від атаки через підміну прошивки або вставку шкідливих фрагментів у код. Зважаючи на обмежені ресурси таких пристроїв – невеликий обсяг пам’яті, обмежене енергоживлення, низька обчислювальна потужність – необхідно обирати криптографічні функції, які є не лише стійкими, але й адаптованими до таких обмежень. Аналіз результатів тестувань на платформах STM32F0, ESP8266, AVR та RISC-V показує, що BLAKE2s забезпечує найкращий баланс між енергоспоживанням, швидкістю та обсягом зайнятої пам’яті, тому пропонується як базовий хеш-механізм у моделі. У випадках, коли апаратно підтримується SHA-256, доцільно використовувати його з міркувань відповідності міжнародним стандартам, попри дещо гіршу енергоефективність. У ситуаціях, коли потрібно потокове хешування, а розмір повідомлення перевищує 1 КБ, рекомендовано використовувати K12 – варіант Keccak із підтримкою фрагментації, що знижує затримку на обробку вхідних даних.

Порівняльна характеристика функцій включає не лише суб’єктивні метрики (зручність реалізації, бібліотечна підтримка), а й об’єктивні, які ґрунтуються на

математичному моделюванні спожитої енергії E та часу виконання T. Зокрема, середня енерговитратність хеш-функції розраховується за формулою (2.2).

$$E = I_{avg} \cdot V \cdot T \tag{2.2}$$

де I_{avg} – середній струм, який споживає мікроконтролер під час виконання функції (у амперах), V – напруга живлення пристрою (у вольтах), T – час виконання обчислення хешу одного блока даних (у секундах).

Для 32-бітного контролера STM32 із живленням 3.3 В і струмом 8 мА, функція BLAKE2s, яка хешує 1 КБ за 2.3 мс, має енерговитратність:

$$E_{BLAKE2s=}0.008 \cdot 3.3 \cdot 0.0023 = 60.72\mu J$$

У порівнянні з SHA-256, який хешує той самий обсяг за 4.6 мс і споживає 11 мА:

$$E_{SHA-256=}0.011 \cdot 3.3 \cdot 0.0046 = 166.98\mu J$$

Отримані результати підтверджують, що на мікроконтролері STM32F103, BLAKE2s виконується у 2 рази швидше та знижує енергоспоживання більш ніж на 40% у порівнянні з SHA-256. На ESP32-S3, завдяки підтримці апаратного прискорення для SHA-256, відмінність менш виражена, однак BLAKE2s зберігає перевагу при потоковому хешуванні. На платформі Longan Nano алгоритм K12 показав кращу продуктивність для довгих повідомлень за рахунок архітектурної оптимізації під RISC-V.

Це дозволяє зробити висновок, що використання BLAKE2s у сенсорних модулях сприяє зниженню загального енергоспоживання при збереженні необхідної криптографічної стійкості, що критично важливо для пристроїв, які живляться від батареї або мають нестабільне живлення.

У межах розробки моделі захисту інформації на рівні вбудованих пристроїв важливим компонентом є перевірка автентичності коду шляхом цифрового підпису. Цей процес унеможливлює запуск шкідливого або зміненого коду на рівні мікроконтролера, навіть якщо зловмисник має фізичний доступ до пристрою або можливість заміни прошивки через OTA-канал. Серед численних

алгоритмів підпису, що існують на сьогодні, найкращі результати надають ECDSA (Elliptic Curve Digital Signature Algorithm) та Ed25519, які підтримують високий рівень безпеки за умови короткої довжини ключів, що важливо для економії пам’яті.

ECDSA базується на еліптичних кривих і дозволяє реалізовувати електронний підпис з довжиною ключа всього 256 біт, що є еквівалентним до 3072-бітного RSA за криптостійкістю. Алгоритм вимагає менше обчислень, а отже й менше енергії, ніж класичні підходи. Формально процес перевірки підпису за допомогою ECDSA полягає у верифікації того, що (2.3).

$$r = (g^u \cdot Q^v mod\ p)_x\ mod\ n \tag{2.3}$$

де r – частина підпису, g – генератор групи, Q – публічний ключ, $u = z/s\ mod\ n, v = r/s\ mod\ n$, z – хеш повідомлення, s – друга частина підпису.

Ed25519, у свою чергу, побудована на кривій Curve25519 і спроектована спеціально для вбудованих систем. Вона демонструє надзвичайно швидке виконання операцій підпису й перевірки (до 10 разів швидше за ECDSA), а також високу стійкість до атак через побічні канали. Ed25519 працює у фіксованому розмірі підпису (64 байти), що полегшує реалізацію у пристроях з жорсткими обмеженнями по пам’яті та продуктивності.

Енергоспоживання перевірки підпису оцінюється за аналогічною формулою (2.4).

$$E_{sign} = I_{sign} \cdot V \cdot T_{sign} \tag{2.4}$$

Для ECDSA зі швидкістю перевірки 6.5 мс при 10 мА:

$$E_{ECDSA=}0.01 \cdot 3.3 \cdot 0.0065 = 241.5\mu J$$

Для Ed25519 зі швидкістю перевірки 1.1 мс при 8 мА:

$$E_{Ed25519=}0.008 \cdot 3.3 \cdot 0.0011 = 29.04\mu J$$

Для підтвердження ефективності реалізації цифрового підпису, було проведено серію вимірювань на платах ESP32-S3 та STM32F103. На ESP32-S3 Ed25519 показала середній час перевірки підпису ~1.1 мс, тоді як на STM32 –

близько 3.7 мс, що свідчить про хорошу адаптивність алгоритму до обмежених систем. В обох випадках рівень використаної оперативної пам’яті не перевищував 2 КБ, а розмір коду – 6 КБ, що дозволяє використовувати підпис навіть на пристроях із жорсткими обмеженнями.

Отже, для енергообмежених пристроїв, що функціонують у бездротових мережах або у важкодоступних місцях, рекомендовано реалізовувати цифровий підпис на основі Ed25519, зважаючи на її високу ефективність, компактність реалізації, і захист від атак на сторонні канали. Це забезпечує надійну автентифікацію програмного забезпечення в рамках загальної моделі захисту сенсорних модулів і вбудованих пристроїв.

2.2.3. Модель загроз і механізмів реагування на етапі завантаження прошивки

У сенсорних вузлах та вбудованих системах із обмеженими ресурсами критично важливим етапом є процес завантаження прошивки, оскільки саме на цьому рівні здійснюється закладення довіри до подальших компонентів програмного забезпечення. Порушення цілісності bootloader або основної прошивки може призвести до повної компрометації системи. Як зазначено у попередніх підрозділах, модель захисту таких пристроїв передбачає поетапну перевірку цілісності та автентичності, однак ефективність цих перевірок залежить від розуміння типів загроз, властивих саме цьому етапу.

Типовими атаками на етапі завантаження є підміна коду прошивки або bootloader, атаки повторного використання (reuse attack), downgrade-атаки із завантаженням попередніх вразливих версій, а також фізичне втручання в апаратне середовище мікроконтролера. Пристрої, які працюють у ворожому середовищі, зокрема сенсорні вузли без безпечного периметру, схильні до спроб модифікації їхньої пам’яті програм або порушення послідовності завантаження.

Bootloader, як корінь довіри, є першою інструкцією, яку виконує мікроконтролер після перезавантаження. Компрометація цього компонента веде до критичного порушення моделі безпеки. Одним з поширених типів атак є заміна bootloader – bootloader replacement attack – коли зловмисник перезаписує законний завантажувач на власний, який імітує очікувану поведінку, але приховано змінює контроль над пристроєм. Ще однією вразливістю є модифікація таблиці векторів переривань, що дозволяє змінити обробку винятків, перехопити керування або змінити шлях виконання коду. Крім того, відкриті інтерфейси налагодження, такі як SWD, JTAG або завантаження через UART у boot mode, можуть дозволити перепрошивку пам’яті без автентифікації, якщо не вимкнені апаратно. У разі успішної атаки на bootloader порушується довіра до будь-якого наступного коду, включаючи верифікацію прошивки, автентичність публічних ключів і навіть функції шифрування/дешифрування.

Прошивка, що виконується після bootloader, також є ціллю атак. Одним із найнебезпечніших сценаріїв є підміна образу прошивки у Flash-пам’яті – як шляхом фізичного доступу до мікросхеми, так і через канал оновлення прошивки Over-the-Air (OTA), зокрема при слабко захищеному з’єднанні або відсутності перевірки цифрового підпису. Downgrade-атаки, що полягають у навмисному завантаженні застарілої, але легітимно підписаної прошивки, можуть відкривати шлях до використання відомих вразливостей попередніх версій. Ще однією проблемою є можливість повторного використання прошивок – clone attack, коли прошивка з одного пристрою переноситься на інший, знеособлюючи систему автентифікації та ідентифікації.

Ці вектори атак особливо актуальні для сенсорних модулів з обмеженими ресурсами, які рідко мають апаратні модулі захисту (TrustZone, TPM, HSM) і вимагають полегшених, але надійних програмних механізмів. Подальші розділи подають конкретні моделі верифікації, реалізації цифрових підписів та структури boot-секвенції з математичним обґрунтуванням та прикладами коду, що

дозволяють побудувати надійну архітектуру навіть на мікроконтролерах STM32F103 або ESP32-S3.

Загрози на етапі завантаження є фундаментальними, оскільки саме під час ініціалізації системи формується корінь довіри (Root of Trust). Сенсорні вузли, позбавлені апаратних засобів захисту (TPM, TrustZone), особливо вразливі до атак, які впливають на завантажувач (bootloader), вектор переривань, або прошивку. На рисунку 2.6 представлено схему завантаження в архітектурі, описаній у підрозділах 2.2.1–2.2.2.

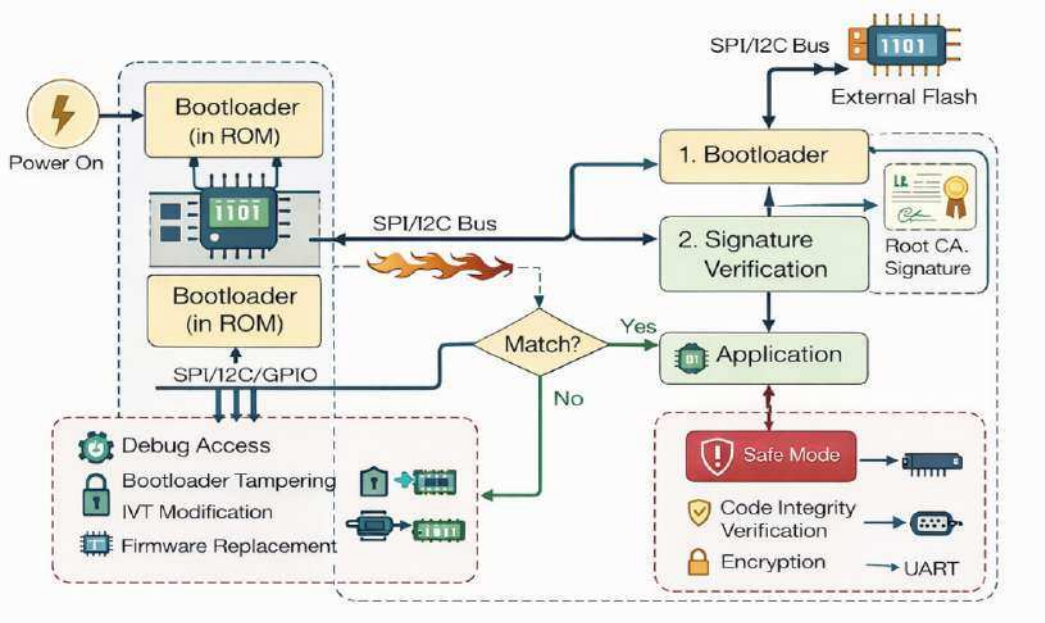

Рисунок 2.6 – Схема ініціалізації сенсорного вузла: bootloader, перевірка підпису, передача керування

На цьому етапі можна виділити три основні групи атак:

1. Атаки на bootloader – bootloader replacement attack – перезапис завантажувача зміненим кодом; модифікація таблиці векторів переривань (IVT tampering) – зміна адрес обробників; debug-атаки через SWD/JTAG/UART – прямий доступ до пам’яті.
2. Атаки на прошивку – firmware image replacement – підміна образу у Flash або OTA; downgrade-атака – завантаження старішої підписаної версії; clone attack – копіювання прошивки між пристроями без зміни UID.

3. Фізичні атаки – зняття зовнішньої Flash, зчитування OTP-блоків, glitching.

В таблиці 2.2 наведено відповідність атак до елементів архітектури сенсорного вузла.

Таблиця 2.2. Класифікація загроз на етапі завантаження та відповідність компонентам архітектури

Тип атаки	Ціль атаки	Механізм впливу	Компонент
Bootloader replacement	Bootloader	Перезапис коду через незахищений інтерфейс	Boot ROM / Flash
Модифікація IVT	Bootloader	Зміна адрес переривань	IVT у Flash
OTA downgrade	Прошивка	Завантаження легітимної, але старої версії	OTA-канал
Flash cloning	Ідентифікація пристрою	Копія прошивки без унікального зв’язку	Application / Flash ID
Debug access	Увесь пристрій	SWD/JTAG/UART з доступом до пам’яті	Debug-інтерфейси

Описані у попередніх підрозділах загрози підкреслюють необхідність інтеграції у boot-секвенцію криптографічних механізмів, здатних забезпечити як автентичність джерела прошивки, так і цілісність її вмісту. Центральну роль у цьому процесі відіграє хеш-функція, яка дозволяє отримати унікальний контрольний відбиток (digest) прошивки, що далі підписується за допомогою приватного ключа та перевіряється пристроєм за допомогою публічного ключа, вбудованого в bootloader. Таким чином, реалізується механізм цифрового підпису, що запобігає підміні firmware-образу, атакам повторного використання та downgrade-атакам. Це відповідає криптографічній схемі, поданій у п. 2.2.2.

Нехай прошивка F має розмір n байт. Хеш-функція $H(F) \to h \in \{0,1\}$, де l – довжина хешу (наприклад, l =256 біт для SHA-256). Цей хеш підписується приватним ключем (2.5).

$$\sigma = Sign_{K_{priv}}(H(F)) \tag{2.5}$$

Під час завантаження bootloader перевіряє справжність (2.6).

$$Verify_{K_{pub}}(H(F), \sigma) == true \tag{2.6}$$

Цей механізм протидіє підміні прошивки, reuse-атакам та downgrade-завантаженням, якщо версія старої прошивки не проходить перевірку лічильника оновлень (2.7).

$$v_{new} > v_{stored} \Longrightarrow allow\ update \tag{2.7}$$

де v_{stored} зберігається у незалежній пам’яті (EEPROM, RTC memory).

Апаратура забезпечує незмінюваний корінь довіри за рахунок eFuse або OTP-пам’яті. Наприклад, у STM32 eFuse містить публічний ключ K_{pub}, який не можна змінити без апаратного знищення. На рис. 2.7 представлено схему розміщення критичних елементів у пам’яті.

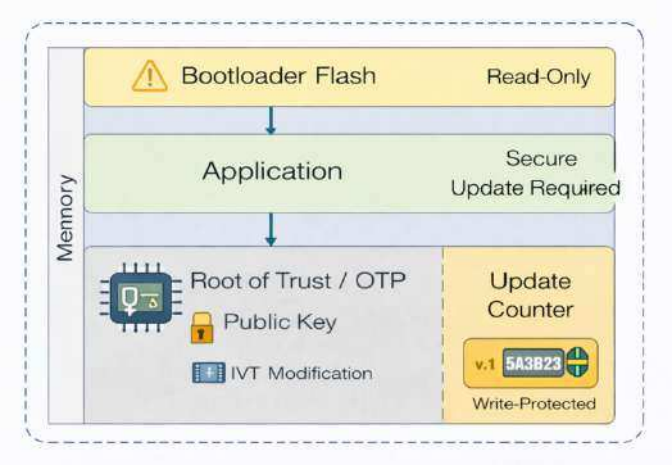

Рисунок 2.7 – Розмежування пам’яті: bootloader, application, eFuse, update counter

Пам’ять поділяється на регіони, де доступ до кожного строго регламентується політиками захисту:

- Bootloader – тільки для читання;

- Application – модифікована OTA, але лише після перевірки підпису;
- Update counter – захищена область, доступна лише bootloader'у.

Модель реакції пристрою у разі виявлення атаки включає кілька сценаріїв:

1. Збій перевірки підпису – блокування запуску; активація OTA-режиму для оновлення.
2. Індикація помилки через GPIO/LED.
3. Виявлення downgrade-атаки – відмова в оновленні; перехід у safe mode з обмеженим функціоналом.
4. Компрометація bootloader – перехід у режим очікування втручання оператора; повна блокада пристрою (secure brick).

В енергозалежних сенсорних вузлах реакція оптимізується шляхом переходу у deep sleep або часткове відновлення лише OTA-підсистеми для повторної перевірки (рис. 2.8).

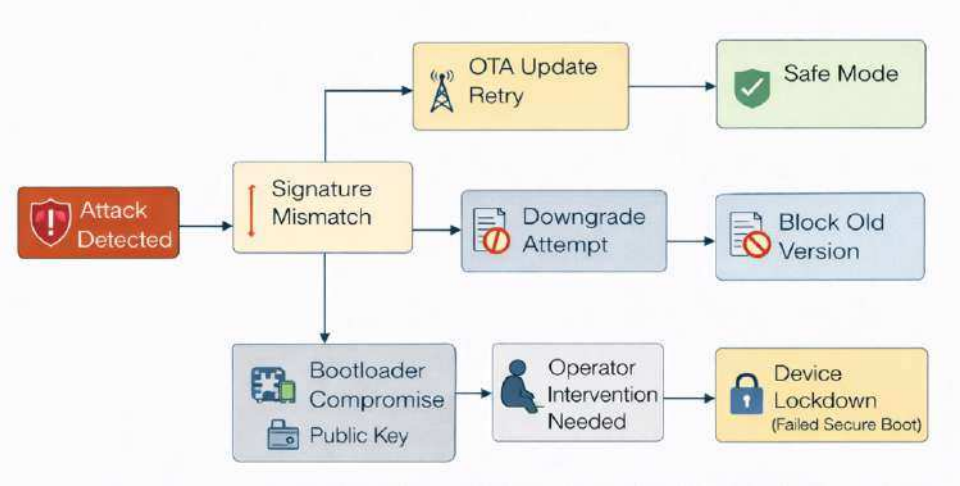

Рисунок 2.8 – Сценарій поведінки пристрою при виявленні атаки

Додатковим елементом захисту виступає апаратно незмінюваний bootloader – наприклад, однократне програмування OTP-блоків пам'яті або використання eFuse (в STM32, ESP32, RISC-V Longan Nano). У такі блоки записується публічний ключ або хеш сертифікованого bootloader, що не може бути перезаписаний або змінений без фізичного пошкодження мікросхеми. Це

виключає можливість підміни кореня довіри, навіть у разі отримання повного доступу до зовнішньої Flash-пам’яті пристрою.

Ще одним контрзаходом є перевірка версій прошивки або лічильників оновлень (update counters), збережених у незалежній пам’яті (наприклад, у секції EEPROM або RTC memory). Перед оновленням bootloader зчитує поточну версію та блокує оновлення у разі спроби завантажити старішу версію, навіть якщо вона підписана правильним ключем. Це дозволяє протидіяти атакам типу downgrade, які використовують архіви попередніх версій.

Архітектурно система захисту також спирається на чітке розмежування областей пам’яті: bootloader розміщується в окремій flash-секції, до якої заборонено доступ на запис із боку користувацької програми (application firmware). У STM32F1, наприклад, для цього використовується механізм захисту сторінок пам’яті (Write Protection, Read-Out Protection), що активується під час виробництва або першого запуску пристрою. У ESP32-S3 подібна функція реалізується через eFuse-політики.

У випадку невдалої верифікації цифрового підпису або порушення хешу прошивки система має реагувати негайно. Сценарії реакції можуть варіюватися залежно від критичності застосування: блокування запуску основної прошивки, переведення пристрою у безпечний режим з обмеженим функціоналом (safe mode), перехід до резервної копії прошивки (dual firmware bank) або індикація помилки через інтерфейс користувача. Наприклад, при виявленні невідповідності хешу на етапі bootloader пристрій не переходить у виконання прошивки, натомість активується механізм OTA-оновлення або затримується запуск до інтервенції оператора.

Для сенсорних вузлів, які часто працюють на акумуляторах або з обмеженим енергоспоживанням, критичною є мінімізація витрат енергії навіть у разі помилок. Тому реакція системи має бути оптимізована: наприклад, пристрій

переходить у режим глибокого сну після фіксації атаки з періодичним пробудженням для повторної перевірки OTA-доступності оновлення.

Таким чином, запропоновані механізми забезпечують не лише базовий рівень автентичності та цілісності, але й відповідають вимогам безпеки, сформульованим у п. 2.1, зокрема – ізоляції компонентів, уникнення повторного використання та надійної реакції на інциденти. Це формує нижній рівень багаторівневої моделі захисту сенсорних вузлів, на який спираються подальші криптографічні механізми, що адаптуються до обмежених ресурсів, як буде розглянуто в п. 2.3.

2.2.4. Оцінка стійкості моделі до атак на етапі ініціалізації пристрою

Етап ініціалізації програмного забезпечення у сенсорних вузлах є одним із найбільш критичних з точки зору інформаційної безпеки, оскільки саме на цьому етапі формується корінь довіри системи та визначається програмний код, який буде виконуватися надалі. Порушення цілісності bootloader або завантажуваної прошивки може призвести до повної компрометації пристрою незалежно від механізмів захисту, реалізованих у прикладному програмному забезпеченні.

У підрозділах 2.2.1–2.2.3 було розглянуто архітектуру Secure Boot, механізми криптографічної перевірки цілісності програмного коду та використання цифрових підписів для підтвердження автентичності прошивки. Для підтвердження ефективності запропонованої моделі необхідно оцінити її здатність протидіяти атакам, характерним для етапу завантаження програмного забезпечення.

Нехай множина можливих атак на етапі ініціалізації сенсорного вузла задається як $A = \{a_1, a_2, \dots, a_n\}$, де a_i – окремий тип атаки.

Розглянемо шість основних атак, характерних для вбудованих сенсорних пристроїв $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$, де:

- a_1 – підміна bootloader;

- a_2 – модифікація прошивки;
- a_3 – downgrade-атака;
- a_4 – клонування або повторне використання прошивки;
- a_5 – несанкціонований доступ через debug-інтерфейси;
- a_6 – підміна прошивки через OTA-канал.

Для кожної атаки вводиться бінарна функція протидії (2.8).

$$f(a_i) = \begin{cases} 1, \text{якщо модель містить мехінзм протидії атаці } a_i \\ 0, \text{якщо ефективного механізму протидії немає} \end{cases} \quad (2.8)$$

Тоді інтегральний показник стійкості системи до атак на етапі ініціалізації можна визначити як частку атак, для яких реалізовано механізм протидії (2.9).

$$R = \frac{1}{n}\sum_{i=1}^{n} f(a_i) \quad (2.9)$$

де n – кількість розглянутих атак.

Відповідність між атаками та механізмами захисту, реалізованими у запропонованій моделі, наведено у таблиці 2.3.

Таблиця 2.3 – Відповідність атак та механізмів протидії

Атака	Основний механізм протидії
Підміна bootloader	захищена область Bootloader та корінь довіри (OTP / eFuse)
Модифікація прошивки	криптографічне хешування та перевірка підпису
Downgrade-атака	контроль версії та лічильник оновлень
Клонування прошивки	прив’язка ключового матеріалу до пристрою
Debug-доступ	апаратні обмеження доступу
Підміна OTA-оновлення	перевірка цифрового підпису

У типовій реалізації Secure Boot, яка використовує лише перевірку хеш-значення або цифрового підпису прошивки, ефективна протидія забезпечується лише для атак модифікації прошивки (a_2) та підміни OTA-оновлення (a_6).

Отже, $f(a_2) = 1, f(a_6) = 1$ для інших атак $f(a_1) = f(a_3) = f(a_4) = f(a_5) = 0$. Обчислимо інтегральний показник стійкості системи до атак за формулою (2.9) – $R_{basic} = \frac{2}{6} \approx 0.33$.

У запропонованій моделі додатково реалізовано:

- захищене зберігання кореня довіри;
- контроль версії прошивки;
- апаратний захист bootloader;
- сценарії безпечної реакції.

Це дозволяє забезпечити протидію атакам a_1, a_2, a_3, a_4, a_6. Тепер обчислимо інтегральний показник стійкості системи до атак у запропонованій моделі за формулою (2.9) – $R_{proposed} = \frac{5}{6} \approx 0.83$.

Отримані результати показують, що запропонована модель перекриває більшість атак, характерних для етапу ініціалізації сенсорних вузлів. Порівняно з базовою реалізацією Secure Boot, яка забезпечує захист лише від модифікації прошивки, запропонована архітектура додатково дозволяє протидіяти атакам підміни bootloader, downgrade-атакам та повторному використанню прошивки.

Кількісна оцінка показує, що запропонована модель забезпечує суттєве підвищення стійкості сенсорних вузлів до атак на етапі ініціалізації, оскільки кількість атак, для яких реалізовано ефективні механізми протидії, збільшується з двох до п’яти.

2.3. Алгоритм адаптивного вибору криптографічного захисту

У вбудованих системах, зокрема в сенсорних вузлах Інтернету речей, вибір криптографічних механізмів обмежується жорсткими апаратними ресурсами, що накладає значні обмеження на складність алгоритмів, обсяг пам’яті, енергоспоживання та час виконання криптографічних операцій. Крім того, актуальні загрози, описані в пп. 2.2.1–2.2.3, вимагають не лише базових засобів

шифрування, а й підтримки автентифікації, цілісності даних та стійкості до атак повторного використання. Таким чином, постає потреба у цілеспрямованому підході до вибору, оцінки та адаптації криптографічних рішень відповідно до специфіки обраної апаратної платформи [175].

На відміну від класичних алгоритмів, таких як AES-128 або RSA-2048, полегшені (lightweight) криптографічні алгоритми розроблено спеціально для умов обмежених ресурсів, де пам’ять, тактова частота і енергобюджет обмежені [176]. Проте ефективність того чи іншого алгоритму не є універсальною і залежить від конкретного типу мікроконтролера (ARM Cortex-M3 у STM32F103 або Xtensa у ESP32-S3), наявності апаратних прискорювачів, обраної операційної системи (FreeRTOS або bare-metal) та типу з’єднання (BLE, Wi-Fi, LoRa).

Узагальнюється підхід до адаптації криптографічного ядра відповідно до системних умов вбудованих платформ. Процедура адаптації ґрунтується на послідовності дій, що охоплює три ключові етапи: визначення релевантних критеріїв вибору з урахуванням апаратних і програмних обмежень, кількісну оцінку продуктивності та криптографічної стійкості кандидатних алгоритмів, а також інтеграцію обраного механізму у цільову платформу з урахуванням особливостей середовища виконання та архітектури системи.

На рисунку 2.9 представлена загальна логіка процедури вибору криптографічного механізму: після аналізу цілей захисту та архітектурних обмежень формується набір критеріїв.

До таких критеріїв належать, зокрема, максимальний розмір програмного коду, витрати енергії та рівень криптографічної стійкості. Далі виконується багатокритеріальна оцінка можливих алгоритмів на основі експериментальних даних або результатів моделювання. Після цього здійснюється адаптація обраного алгоритму до цільової платформи. Така адаптація включає оптимізацію реалізації, інтеграцію у вигляді бібліотеки або включення алгоритму до послідовності завантаження системи. Запропонований підхід дозволяє узгодити

обраний криптографічний механізм з обмеженнями та вимогами проєктованої системи.

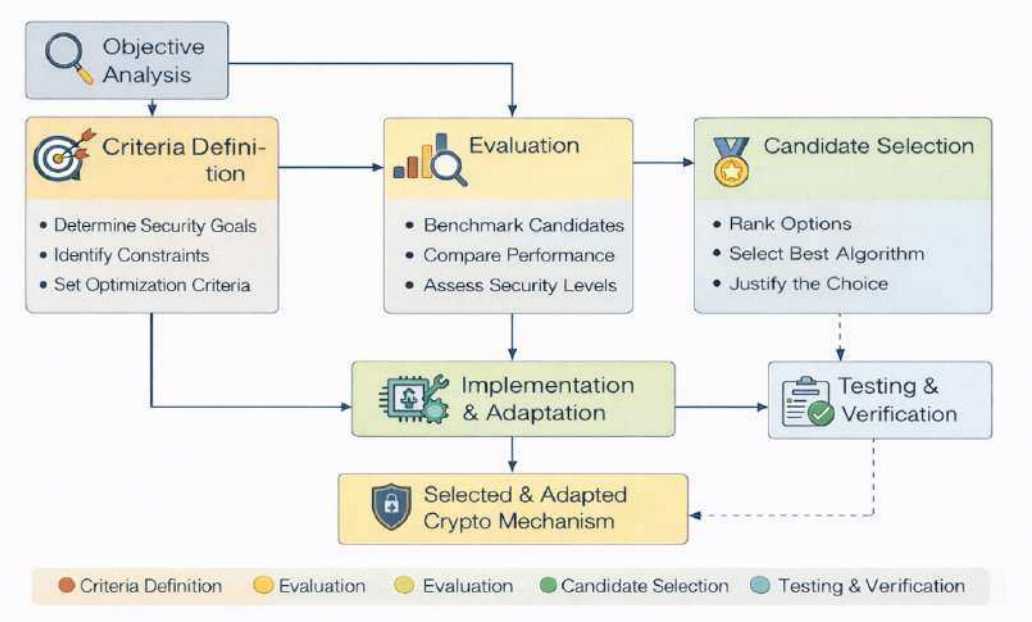

Рисунок 2.9 – Алгоритм адаптивного вибору криптографічного захисту для вбудованої системи

Після етапу впровадження виконуються додаткові перевірки, зокрема оцінка продуктивності, порівняння з еталонними реалізаціями, перевірка стійкості до типових атак (наприклад, side-channel атаки, атаки відновлення ключа), а також тестування сумісності з операційною системою. У деяких випадках необхідна адаптація форми API або створення апаратної абстракції (HAL).

2.3.1. Критерії вибору алгоритмів для пристроїв з обмеженими ресурсами

У системах з обмеженими обчислювальними ресурсами, як-от сенсорні вузли в IoT, процес вибору криптографічного алгоритму передбачає аналіз не лише криптографічної стійкості, а й відповідності апаратним обмеженням. Базовими критеріями для відбору кандидатів є обсяг споживаної пам’яті, енергетична вартість криптографічних операцій, пропускна здатність (Throughput) та час виконання окремих функцій. Додатково враховуються

сумісність з обраною мікроархітектурою, доступність бібліотек, підтримка вбудованого програмного середовища та можливість захисту від сторонніх каналів (side-channel resistance).

Розглянемо подання критеріїв у вигляді обмежень та показників ефективності. Нехай:

- M_{code} – обсяг пам'яті програмного коду, байт;
- M_{RAM} – необхідний обсяг оперативної пам'яті, байт;
- E_{op} – енергоспоживання на одну криптографічну операцію, мкДж;
- T_{enc} – середній час шифрування одного блоку, мс;
- T_{dec} – час розшифрування;
- S – рівень криптографічної стійкості (запас у бітах щодо атаки перебором).

Формується вектор обмежень (2.8).

$$\vec{C} = \begin{cases} M_{code} \le M_{max} \\ M_{RAM} \le R_{max} \\ E_{op} \le E_{max} \\ T_{enc}, T_{dec} \le T_{max} \\ S \ge S_{min} \end{cases} \quad (2.8)$$

де M_{max}, R_{max}, E_{max}, T_{max}, S_{min} – граничні значення, встановлені згідно з профілем платформи (наприклад, STM32F103 має 20 кБ RAM та 64 кБ Flash).

Доцільно також визначити узагальнений показник ефективності Q для порівняння альтернатив. Один із можливих підходів – нормалізована вага кожного критерію (2.9).

$$Q = w_1 \cdot \frac{M_{code}}{M_{max}} + w_2 \cdot \frac{M_{RAM}}{R_{max}} + w_3 \cdot \frac{E_{op}}{E_{max}} + w_4 \cdot \frac{T_{enc}}{T_{max}} + w_5 \cdot \frac{1}{S} \quad (2.9)$$

де $w_i \in [0,1], \sum w_i = 1$, і значення підбираються залежно від пріоритетів (наприклад, у пристроях з живленням від батареї вагу w_3 слід підвищити).

Крім того, важливим фактором є здатність алгоритму до гнучкого масштабування – можливість зміни довжини ключа або блоку для адаптації до різних рівнів захисту. Важливою є й наявність відкритого коду, відповідність

міжнародним стандартам (ISO/IEC 29192, NIST LWC), а також результати зовнішніх криптоаналізів. У табл. 2.4 наведено приклад оцінки кандидатів з використанням типових параметрів для вбудованих пристроїв.

Таблиця 2.4 Порівняння параметрів lightweight-алгоритмів

Алгоритм	M_{code}, байт	M_{RAM}, байт	E_{op}, мкДж	T_{enc}, мс	S, біт	Підтримка стандартів
AES-128	8100	210	22	0.54	128	ISO 18033-3
SPECK64/128	1800	96	8.1	0.27	96	NIST LWC Candidate
Ascon-128	2050	68	11.3	0.34	128	ISO/IEC 29192-6
GIFT-COFB	1950	85	10.6	0.31	128	NIST LWC Finalist
PRESENT	1400	45	6.2	0.76	80	ISO/IEC 29192-2

Як видно з табл. 2.6, незважаючи на схожий рівень безпеки, різні алгоритми мають різні профілі споживання пам’яті та енергії, що може бути критичним при розміщенні на конкретному чипі. Наприклад, у разі обмежень по Flash більше 4 КБ і потреби у високій швидкодії краще обрати SPECK, однак за вищих вимог до стійкості рекомендовано Ascon-128.

Для порівняння продуктивності полегшених криптографічних алгоритмів на різних мікроконтролерах було проведено експериментальне вимірювання часу шифрування одного блоку даних (64 або 128 біт). Як видно з рисунка 2.10, алгоритми SPECK64/128 та GIFT-COFB демонструють найменший час обробки, що робить їх придатними для систем з критичними вимогами до затримки. Натомість PRESENT має найбільший час шифрування, що знижує його придатність у високочастотних обмінних сесіях.

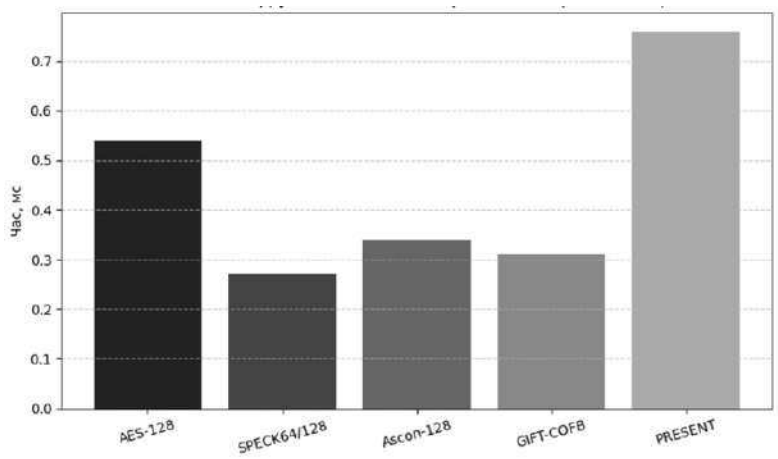

Рисунок 2.10 – Час шифрування одного блоку даних (мс)

Енергоспоживання є критичним показником для сенсорних вузлів, що працюють від автономного живлення. На рисунку 2.11 показано кількість мікроджоулів, необхідних для виконання однієї криптографічної операції. Алгоритми PRESENT та SPECK64/128 забезпечують найнижчі енергозатрати, тоді як AES-128 та Ascon-128 є менш енергоефективними, що обмежує їхнє застосування в енергонезалежних системах.

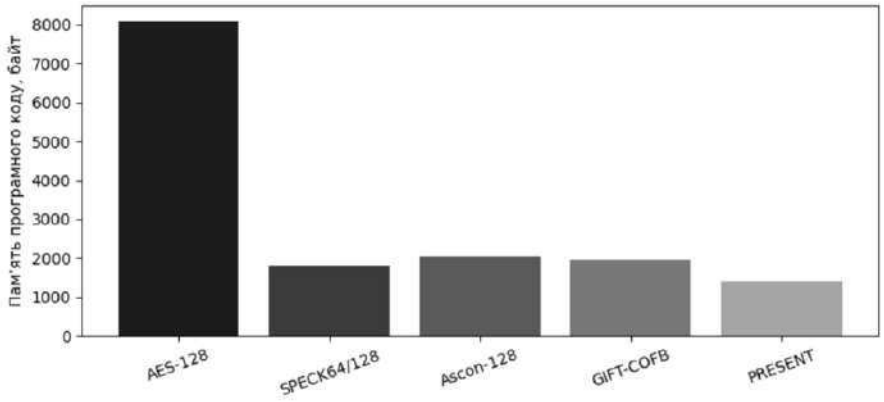

Рисунок 2.11 – Енергоспоживання на одну операцію шифрування (мкДж)

На рис. 2.12 представлено порівняння потреб алгоритмів у пам’яті, яка включає як Flash (ROM), так і оперативну пам’ять (RAM). Цей показник визначає можливість розміщення криптографічного механізму на мікроконтролері з обмеженими ресурсами. Найменше споживання пам’яті характерне для

PRESENT, тоді як AES-128 вимагає у 4–5 разів більше ресурсів, що ускладнює його використання на таких платформах, як STM32F103 або ATTiny85.

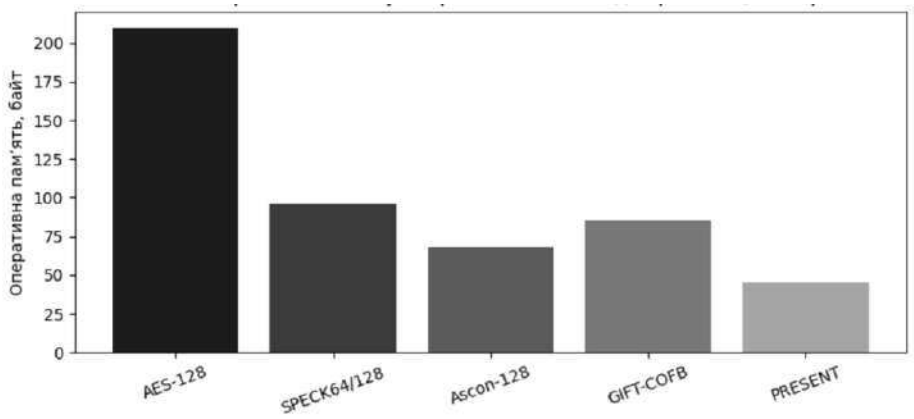

Рисунок 2.12 – Споживання пам’яті (Flash + RAM)

У процесі вибору криптографічного механізму важливу роль відіграє не лише відповідність технічним обмеженням, а й сумісність алгоритму з архітектурними особливостями конкретного мікроконтролера. Наприклад, апаратна підтримка побітових операцій або арифметики в GF(2^n), наявність інструкцій SIMD, прискорення для AES або криптографічного акселератора можуть суттєво впливати на доцільність впровадження певного алгоритму. Для багатьох чипів сімейства STM32 існують оптимізовані реалізації AES або ChaCha20, які використовують апаратну підтримку шифрування, тоді як алгоритми Ascon чи GIFT-COFB потребують оптимізації програмного коду під особливості ядра ARM Cortex-M.

Окремо слід враховувати безпеку проти атак сторонніми каналами (side-channel attacks), таких як аналіз споживання енергії (DPA), електромагнітного випромінювання (EMA), чи аналіз часу виконання (timing attacks). Деякі алгоритми демонструють вищу резистентність до таких атак завдяки своїй побудові (наприклад, Ascon завдяки sponge-конструкції), або підтримці константного часу виконання. Водночас, відсутність апаратної ізоляції (наприклад, захищеної області пам’яті) у мікроконтролерах нижчого класу

накладає додаткові вимоги до вибору реалізації, яка повинна бути написана з урахуванням SCA-митигуючих практик.

Іншим критичним аспектом є підтримка обраного алгоритму у стандартних бібліотеках та фреймворках, з якими працює розробник. Наприклад, інтеграція з середовищем FreeRTOS, Mbed OS або Zephyr передбачає використання готових middleware-компонентів, які можуть мати лише обмежений перелік підтримуваних шифрів. У разі необхідності реалізації з нуля важливо переконатися у доступності перевірених open-source реалізацій, з належною документацією та проходженням верифікації в рамках криптоаналітичних конкурсів.

Остаточний вибір алгоритму залежить не лише від його властивостей, а й від контексту використання: тип даних, тривалість сеансу зв’язку, наявність механізмів поновлення ключів і стратегії оновлення прошивки. Наприклад, для короткоживучих сенсорів у моніторингу навколишнього середовища прийнятною є реалізація з мінімальним енергоспоживанням, навіть ціною зниження пропускної здатності, тоді як для вузлів керування у медичних або промислових пристроях першочерговим є високий рівень безпеки та стійкість до атак з боку локального фізичного доступу. Таким чином, формування профілю захищеності має передувати вибору алгоритму і визначати вагу кожного критерію в узагальненій моделі прийняття рішень.

2.3.2. Оцінка параметрів обраних криптографічних рішень

Після попереднього етапу відбору алгоритмів за релевантними критеріями виконується їх детальна кількісна оцінка з урахуванням особливостей цільової обчислювальної платформи. Основною метою цього етапу є отримання об’єктивних показників ефективності – як у плані продуктивності, так і в аспекті споживання апаратних ресурсів. Це дозволяє обґрунтовано обрати алгоритм, що найкраще відповідає умовам функціонування вбудованої системи.

Ключовими метриками для оцінювання продуктивності криптографічних механізмів у вбудованих пристроях є:

- час виконання криптографічних операцій (encryption/decryption latency);
- кількість тактів процесора (clock cycles per block);
- споживання енергії на одне шифрування/дешифрування (energy per operation);
- використання пам’яті – як оперативної (RAM), так і постійної (Flash/ROM);
- пропускна здатність (throughput) за фіксованих умов (наприклад, 128 біт/сек);
- масштабованість та адаптивність до різних конфігурацій (кількість раундів, довжина ключа).

Для вимірювання зазначених характеристик використовуються два підходи: експериментальні тести на фізичних пристроях (наприклад, STM32F407VG та ESP32-S3) та програмне профілювання у віртуалізованому середовищі (наприклад, QEMU, Renode, або STM32CubeIDE). В обох випадках застосовується векторний набір тестів, що охоплює типові сценарії обміну даними в межах IoT-системи: одноразове шифрування фрейму, пакетна обробка, генерація автентифікаційного тега тощо.

Для забезпечення точності вимірювань енергоспоживання використовуються енергетичні аналізатори (наприклад, STM32 Power Shield або Espressif ESP-Prog), які дозволяють виміряти спожиту енергію під час виконання шифрувального циклу з точністю до мікроджоуля. При цьому результати нормалізуються до кількості оброблених бітів (наприклад, nJ/bit), що дозволяє порівнювати алгоритми між собою незалежно від їх блочної структури.

Особливу увагу приділено стабільності часу виконання**.** Наприклад, алгоритми зі змінною кількістю операцій (наприклад, залежних від значень вхідного повідомлення) є менш придатними для безпечного використання у вбудованих системах, оскільки створюють передумови для таймінгових атак**.**

Тому важливою частиною оцінювання є верифікація того, що реалізація працює у константному часі (constant-time execution).

У таблиці 2.5 наведено результати вимірювань для обраних алгоритмів (Ascon, GIFT-COFB, QARMA, AES-128) при реалізації на STM32F407 під FreeRTOS. Дані містять як продуктивні, так і енергетичні характеристики, а також показники використання пам’яті.

Таблиця 2.5 – Порівняння криптографічних механізмів на STM32F407

Алгоритм	Час шифрування блоку (мкс)	RAM (байт)	Flash (байт)	Енергія (нДж/біт)	Стійкість до атак часу
Ascon-128a	47,2	1600	8300	32,4	Висока
GIFT-COFB	38,9	900	7600	28,1	Середня
QARMA-64	42,7	1300	7200	30,7	Висока
AES-128 (SW)	63,5	1800	9800	44,8	Низька

На основі таких даних виконується формування рейтингової оцінки з урахуванням вагових коефіцієнтів для кожного критерію. Наприклад, для автономних пристроїв з живленням від батареї вирішальним критерієм є енергоспоживання, тоді як для систем реального часу – затримка обробки. Таким чином, підсумковий рейтинг може змінюватись залежно від умов експлуатації, що забезпечує адаптивність підходу.

2.3.3. Впровадження алгоритмів у програмне середовище пристрою

Реалізація обраного криптографічного алгоритму в умовах вбудованої системи є критичним етапом, який вимагає адаптації до особливостей обраного середовища виконання – FreeRTOS, Zephyr чи bare-metal. Успішне впровадження забезпечується завдяки сумісності з інтерфейсами апаратного доступу (HAL/LL), можливості використання DMA для оптимізації роботи з пам’яттю, зменшенню споживання енергії та використанню стека, а також реалізації потокового обчислення хеш-функцій.

У середовищі FreeRTOS та Zephyr підтримується розділення задач на окремі потоки з пріоритетами, що дозволяє запускати перевірку підпису або обчислення хешу в окремому task без блокування основного циклу. При цьому, з метою зниження навантаження на процесор, рекомендується використовувати DMA-передачу при читанні прошивки з Flash у буфер, а також обчислення хешу блоками. На платформі STM32 це реалізується через LL DMA API із підтримкою алгоритмів типу BLAKE2s, які не потребують збереження у великому буфері повного повідомлення.

Для bare-metal реалізацій, де відсутня абстракція операційної системи, код повинен бути максимально оптимізований щодо використання регістрів, уникати динамічного виділення пам’яті й реалізовувати цикл перевірки без затримок. Такий підхід потребує ручного налаштування контролерів переривань, SPI/UART-інтерфейсів, а також обробки помилок у реальному часі. У таблиці 2.4 проведено порівняння типових характеристик реалізації одного й того самого алгоритму (Ascon-128) у середовищах bare-metal, FreeRTOS і Zephyr на STM32F407.

Серед проаналізованих алгоритмів було обрано Ascon-128 як експериментальну реалізацію з огляду на збалансованість між криптографічною стійкістю, енергоефективністю та витратами пам’яті. Алгоритм входить до переліку фіналістів конкурсу NIST Lightweight Cryptography, має відкриту перевірену реалізацію, підтримує режим AEAD (authenticated encryption with associated data), що дає змогу одночасно забезпечувати конфіденційність і цілісність. Його побудова на основі sponge-конструкції дозволяє адаптувати обчислення до потокової обробки, що особливо важливо для пристроїв із малим обсягом оперативної пам’яті. Крім того, Ascon-128 показує стабільні результати при захисті від атак сторонніми каналами, включно з DPA та аналізом часу виконання, завдяки можливості реалізації у константному таймінгу. Вибір саме

цього алгоритму дає змогу об’єктивно оцінити особливості інтеграції сучасних lightweight-рішень у типові середовища вбудованих систем.

Таблиця 2.6 – Порівняння реалізацій Ascon-128 у різних середовищах

Середовище	Час виконання (мс)	Споживання RAM (КБ)	Стек (Байт)	Кількість рядків коду	Особливості
Bare-metal	3.9	1.4	512	430	Повна оптимізація вручну
FreeRTOS	4.3	2.2	784	520	Інтеграція в окремий task
Zephyr	4.1	2.6	820	610	Використано Zephyr crypto API

Як видно з таблиці 2.4, реалізація у середовищі bare-metal демонструє найменший час виконання за рахунок відсутності абстракцій і контекстних перемикань, однак вимагає найбільших зусиль щодо підтримки й модифікації. FreeRTOS забезпечує кращу масштабованість при мінімальному збільшенні споживання ресурсів, а Zephyr пропонує широку інтеграцію з існуючими драйверами, проте має найбільший розмір коду.

На рисунку 2.13 наведено загальну архітектурну схему інтеграції криптографічного модуля у прошивку вбудованого пристрою, де показано його зв’язок із модулями обробки подій, керування живленням, OTA-оновленням та bootloader.

Побудова такої архітектури дозволяє досягти гнучкості в налаштуванні рівня захисту, а також забезпечити ізольованість криптографічних функцій, що спрощує верифікацію, аудит та оновлення без порушення логіки основної програми. Подальше вдосконалення полягає в автоматичній генерації ключів, адаптації до різних профілів живлення та реалізації механізмів постійного моніторингу коректності виконання (watchdog + контроль хешу).

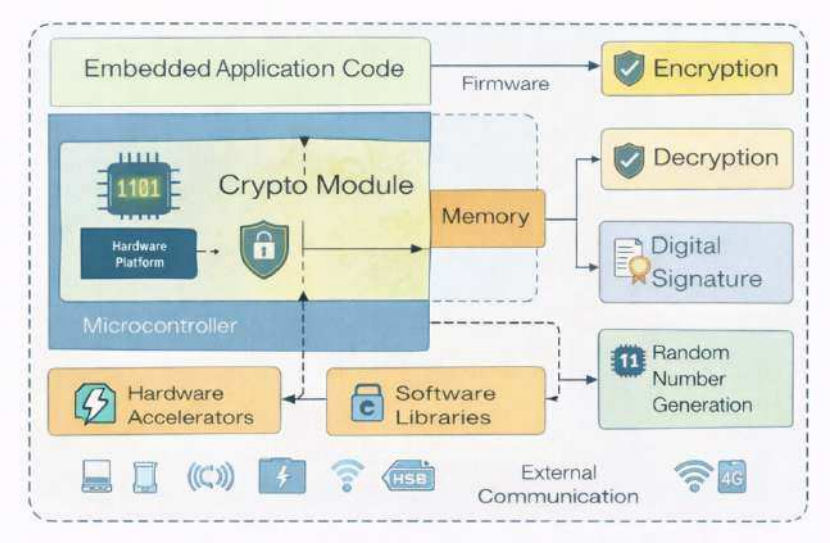

Рисунок 2.13 – Схема інтеграції криптомодуля у прошивку вбудованої системи

2.4. Модель управління криптографічними ключами сенсорного рівня

Криптографічні ключі є основою будь-якої моделі безпеки, орієнтованої на конфіденційність, автентичність і цілісність даних у вбудованих сенсорних пристроях. Для обчислювальних систем з обмеженими ресурсами характерне застосування симетричних алгоритмів шифрування, де однаковий ключ використовується як для шифрування, так і для дешифрування. Проте в умовах побудови динамічних мереж або забезпечення автентичності критично важливих оновлень, таких як firmware або сеансові повідомлення, також використовуються пари відкритого та приватного ключів, зокрема на основі Ed25519 або X25519. У структурі системи безпеки виділяють кілька типів ключів: кореневі (вшиті на етапі виробництва), сеансові (генеруються при встановленні з’єднання), тимчасові (наприклад, для автентифікації OTA-оновлень), та ключі автентифікації повідомлень.

Для шифрування даних у вбудованих пристроях часто застосовуються симетричні ключі, наприклад, у схемах з AES-128 або полегшеними алгоритмами типу SPECK, GIFT, PRESENT. Їх основна перевага – висока швидкодія і низьке

навантаження на ресурси мікроконтролера. Проте для встановлення захищених з'єднань або перевірки підпису в оновленнях прошивки все частіше використовуються асиметричні пари ключів, зокрема Ed25519 або Curve25519, які поєднують компактність ключів і прийнятну швидкість обчислень навіть на малопотужних процесорах.

Окреме місце у структурі ключового захисту займають кореневі ключі Root of Trust, які закладаються у пристрій на етапі виробництва або під час початкової ініціалізації. Вони не змінюються впродовж життєвого циклу пристрою та використовуються для перевірки автентичності прошивки чи компонентів ПЗ, формуючи основу довірчого ланцюга. У підрозділі 2.2.2 розглянуто механізм криптографічного підпису прошивки, де RoT-ключ використовується для перевірки підпису в завантажувачі.

Інтеграція модулів захисту на основі криптографічних механізмів, описаних у попередніх підрозділах, передбачає чітке розділення ролей кожного типу ключів. Кореневі ключі не повинні залишати пристрій і використовуються для верифікації підписів, тоді як сеансові ключі можуть обмінюватися між пристроями за допомогою протоколів захищеного встановлення з’єднання. Кожна взаємодія між вузлами має супроводжуватися механізмами підтвердження володіння ключем, з використанням електронного підпису або обчислення MAC.

2.4.1. Методи генерації ключів у вбудованих пристроях

Ефективна та безпечна генерація криптографічних ключів є критично важливою для побудови довірчого середовища у вбудованих системах. З огляду на обмежені ресурси пристроїв, зокрема сенсорних вузлів у IoT-мережах, генерація ключів має базуватись на легких і водночас надійних механізмах, здатних забезпечити необхідний рівень ентропії та криптографічної стійкості.

Одним із основних підходів є використання апаратного генератора випадкових чисел (True Random Number Generator, TRNG). У сучасних мікроконтролерах, таких як STM32 та ESP32, такі генератори реалізуються на апаратному рівні та доступні через відповідні периферійні модулі. Наприклад, у STM32 TRNG надає інтерфейс до генератора шуму, який забезпечує апаратне джерело ентропії для криптографічних операцій. Аналогічно, ESP32 має вбудований RNG, який дозволяє отримувати випадкові 32-бітні значення напряму з системного API.

Альтернативним або додатковим методом може бути генерація ключів на основі часових токенів, які формуються на основі мікросекундного або мілісекундного таймера під час запуску пристрою або внаслідок подій з непередбачуваним часом настання. Також можливе використання фонових флуктуацій середовища – температури, напруги, затримок при читанні з аналогових входів – як джерела ентропії. Ці методи потребують додаткової обробки для зменшення детермінізму та нормалізації випадкових послідовностей, що особливо важливо у випадку формування довготривалих ключів.

На рисунку 2.14 подано схему генерації пари ключів у сенсорному пристрої, що використовує апаратний TRNG для створення закритого ключа, після чого обчислюється відкритий ключ відповідно до алгоритму Ed25519.

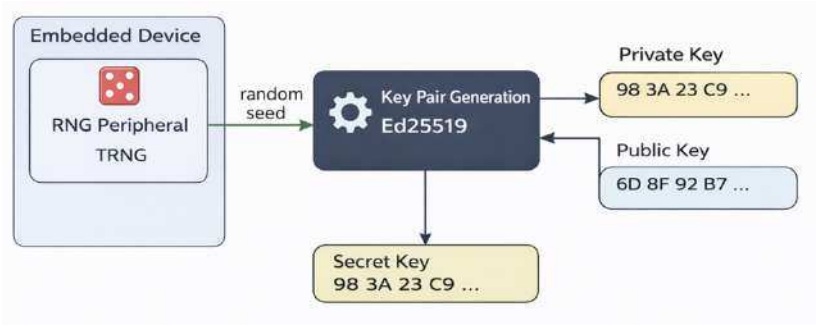

Рисунок 2.14 – Схема генерації пари ключів у сенсорному пристрої з використанням TRNG

У середовищі FreeRTOS реалізація генерації ключової пари Ed25519 може виконуватись у фоновому завданні з використанням API бібліотек monocypher або TweetNaCl, які адаптовані до вбудованих систем. Отримане випадкове значення з TRNG або псевдовипадкова послідовність після обробки ентропійного пулу використовується як приватний ключ, а відповідна функція обчислення відкритого ключа – як частина ініціалізації безпечного з'єднання або формування цифрового підпису. Повний приклад такої реалізації наведено у Лістингу А.2. (Додаток А).

2.4.2. Безпечне зберігання криптографічних ключів

Безпечне зберігання криптографічних ключів є критично важливим аспектом забезпечення стійкості до атак та збереження конфіденційності вбудованих пристроїв. Ключі, що зберігаються у відкритому або недостатньо захищеному вигляді, стають вразливими до компрометації, особливо у випадку фізичного доступу до пристрою або застосування атак через побічні канали.

Існує декілька варіантів зберігання ключів у вбудованих системах:

1. eFuse – електронні запобіжники, які дозволяють одноразовий запис і забезпечують високу стійкість до модифікацій. Вони часто використовуються для зберігання ідентифікаторів пристрою або кореневих ключів.

2. OTP (One-Time Programmable) пам'ять – застосовується для незмінного зберігання ключової інформації, але має обмеження щодо повторного використання.

3. Захищена область Flash-пам'яті – програмно обмежена зона постійної пам'яті з доступом лише з певного контексту або на певних етапах завантаження.

4. SRAM із захистом доступу – використовується переважно для тимчасового зберігання, знищення ключів відбувається при знеструмленні, що забезпечує високий рівень захисту у разі несанкціонованого фізичного втручання.

У таблиці 2.7 наведено порівняння основних способів зберігання криптографічних ключів у вбудованих пристроях.

Таблиця 2.7 – Порівняльна характеристика способів зберігання ключів у вбудованих пристроях

Спосіб зберігання	Стійкість до атак	Можливість перезапису	Захист від побічних каналів	Приклад реалізації
eFuse	Висока	Немає	Високий	ESP32 eFuse, STM32 HUK
OTP	Висока	Немає	Середній	STM32 OTP Memory
Захищена Flash	Середня	Так	Залежить від реалізації	Flash Secure Area
SRAM з захистом	Висока	Так	Високий	STM32 TrustZone RAM

Для додаткового захисту, ключі можуть зберігатися в ізольованих середовищах, таких як TrustZone у STM32L5 або Secure Bootloader Partition у ESP32-S3. Деякі мікроконтролери також підтримують апаратне блокування читання (RDP – Read Protection Level) та реалізують протидію атакам побічними каналами за рахунок апаратної обфускації, додавання шуму або фізичного екранування.

Варто звернути увагу на документацію щодо політик безпеки апаратних платформ. Зокрема, у STM32 використовується механізм TrustZone-M та Secure Boot з можливістю захищеного сховища. У ESP32-S3 передбачено використання eFuse-бітів для блокування доступу до ключів та реалізовано Secure Boot V2 з перевіркою підписів і шифруванням Flash-пам’яті.

2.4.3. Ротація та оновлення ключів у динамічному середовищі

Захист криптографічних ключів у вбудованих пристроях, що функціонують в умовах обмежених ресурсів та нестабільного середовища (втрата живлення, нестійкий зв’язок, перебої в комунікації), потребує впровадження ефективних механізмів ротації та оновлення ключової інформації. Своєчасна ротація ключів

сприяє зменшенню ризику компрометації і забезпечує підтримку довіреного стану системи впродовж її життєвого циклу.

Серед актуальних моделей оновлення криптографічних ключів у вбудованих системах виокремлюють:

- модель ключів епох (epoch keys), в якій для кожного періоду або сеансу використовується окремий ключ, що автоматично оновлюється з настанням нового інтервалу;
- ланцюгова модель еволюції ключів, що базується на послідовній генерації нових ключів з попередніх шляхом застосування односпрямованої функції (наприклад, хеш-функції);
- деревоподібні структури ключів, які дозволяють ієрархічне оновлення ключової інформації з мінімальним впливом на пов'язані пристрої, забезпечуючи масштабованість системи в умовах мережевої динаміки.

На рисунку 2.15 представлено узагальнену модель ланцюгової еволюції ключів із включенням лічильника оновлень, що забезпечує контроль за синхронністю та унеможливлює повторне використання застарілих версій ключів.

У реалізації таких моделей ключові параметри, як лічильник оновлень (update counter) та одноразові значення (nonce), відіграють ключову роль у запобіганні атакам повторного відтворення та синхронізаційним збоям. Використання лічильника гарантує, що новий ключ не може бути повторно скомбінований із попереднім станом системи, навіть у разі втрати з'єднання або відновлення після помилки.

У випадку виявлення компрометації ключа або втрати контексту синхронізації передбачається повторна ініціалізація процедури узгодження ключів із використанням спеціального протоколу аварійного оновлення або резервної процедури розповсюдження кореневого ключа. Для цього вбудовані системи повинні зберігати невеликий запас заздалегідь згенерованих ключових

матеріалів або підтримувати механізм запиту нового ключа від довіреного центру.

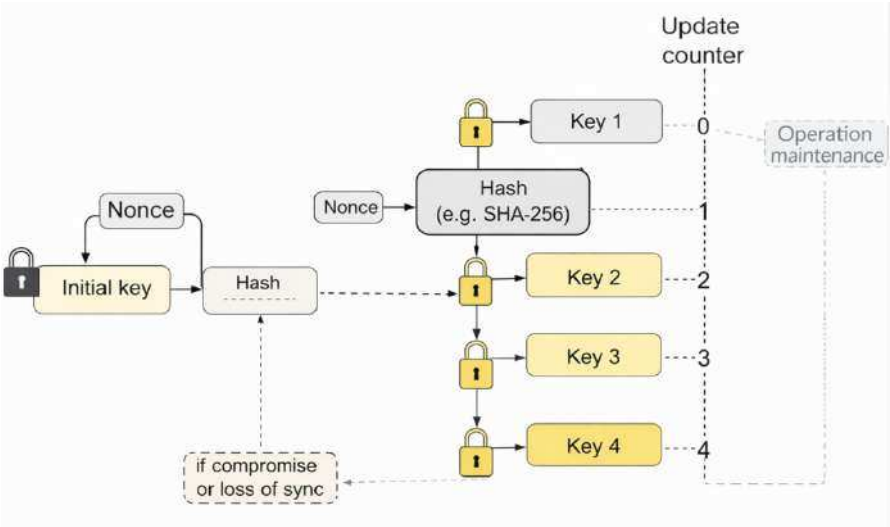

Рисунок 2.15 – Модель еволюції ключів з ланцюговою залежністю та контролем оновлень

У таких середовищах доцільним є впровадження апаратної або програмної перевірки актуальності ключа за допомогою внутрішніх маркерів часу або криптографічних підписів, які підтверджують достовірність та походження оновленої ключової інформації.

2.4.4. Захищена синхронізація ключів між вузлами і сервером

У контексті вбудованих пристроїв з обмеженими ресурсами захищена синхронізація ключів між сенсорними вузлами та центральним сервером або шлюзом має критичне значення для цілісності та конфіденційності обміну даними. Сучасні криптографічні протоколи дозволяють встановлювати спільні ключі без попереднього обміну симетричними секретами, що є важливою вимогою для динамічних IoT-архітектур.

Одним із найпоширеніших підходів до побудови спільного секрету між вузлами є використання протоколу узгодження ключів на основі еліптичних кривих, зокрема Elliptic-Curve Diffie–Hellman (ECDH). У сучасних реалізаціях

дедалі частіше використовуються оптимізовані варіанти на основі кривої Curve25519, такі як X25519, що мають переваги з точки зору продуктивності та стійкості до відомих атак. Як перспективний напрям розвитку розглядається використання постквантових схем узгодження ключів, зокрема на основі граткових криптосистем, які дозволяють зберігати безпеку в умовах появи квантових обчислень.

Верифікація автентичності публічних ключів здійснюється через ієрархічну модель ланцюга довіри. У типовій реалізації публічний ключ пристрою підписується проміжним сертифікатом, який, у свою чергу, верифікується кореневим сертифікатом, збереженим у захищеній області пам’яті шлюзу або сервера. Така модель забезпечує цілісність і недоступність до ключів у разі атак типу «людина посередині».

Оновлення сеансових ключів у динамічному середовищі відбувається за допомогою захищеного каналу з використанням Over-the-Air (OTA) механізму. Алгоритм оновлення включає попередню автентифікацію вузла, генерацію нового ключа на основі алгоритму з ротацією, шифрування оновлення на стороні сервера, підпис оновлення приватним ключем виробника та подальшу перевірку цілісності оновлення на пристрої. Кожен з етапів супроводжується контролем зворотного зв’язку, тайм-аутами на рівні мережевого стека та маркуванням оновлень для запобігання повторному застосуванню.

На рисунку 2.16 наведено послідовність обміну сеансовими ключами між сенсорним пристроєм та шлюзом із використанням X25519 та підписаних сертифікатів, що ілюструє повний цикл автентифікації, обчислення спільного секрету та підтвердження успішної синхронізації.

Після ініціалізації з’єднання сенсорний пристрій генерує пару ключів на основі кривої Curve25519. Публічний ключ пристрою передається шлюзу разом із сертифікатом, підписаним Root of Trust, збереженим у заздалегідь запрограмованій пам’яті шлюзу. Шлюз верифікує сертифікат, перевіряє

автентичність ключа та генерує власну пару ключів, публічна частина якої надсилається у відповідь.

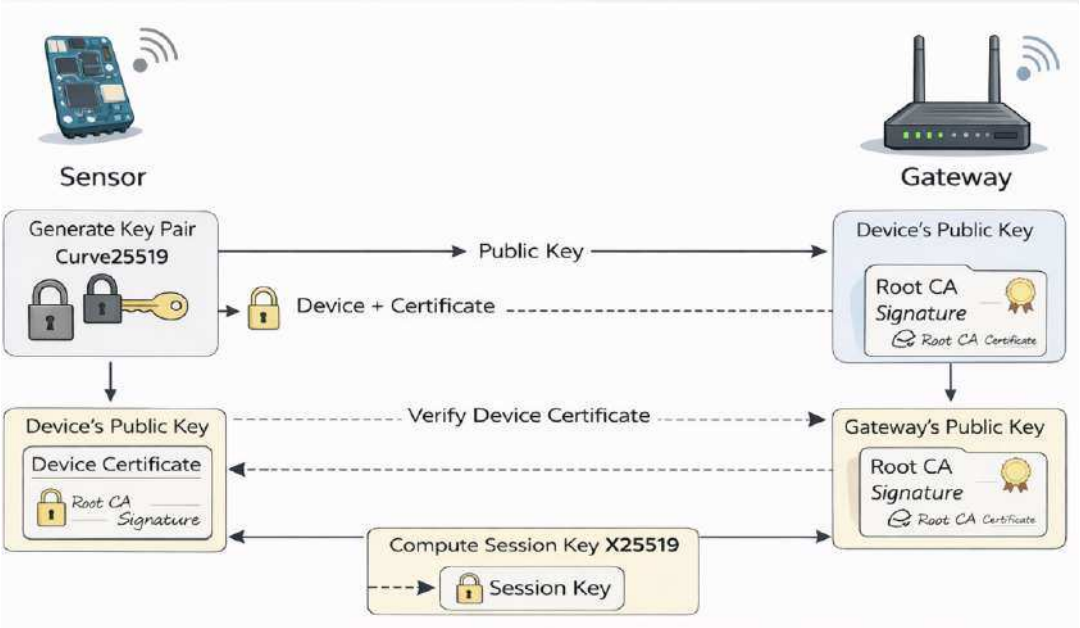

Рисунок 2.16 – Послідовність обміну сеансовими ключами між сенсором і шлюзом із використанням X25519 та сертифікатів автентичності

Обидві сторони незалежно обчислюють спільний сеансовий ключ за допомогою протоколу X25519. Сеансовий ключ не передається мережею, що виключає його перехоплення. Для захисту від атак повторного використання та підміни повідомлень, обмін ключами супроводжується унікальними ідентифікаторами сесії, лічильниками оновлень та часовими мітками.

У разі успішного завершення процедури синхронізації обидві сторони переходять до зашифрованого обміну даними з використанням новоствореного сеансового ключа. Система контролює часовий інтервал дії ключа, кількість переданих пакетів та інші параметри для запуску наступної ротації за визначеними політиками безпеки.

2.4.5. Визначення вимог до керування ключами

У контексті вбудованих систем із обмеженими обчислювальними ресурсами управління криптографічними ключами потребує врахування як апаратних, так і програмних обмежень, а також оцінки ризиків, пов'язаних з їх компрометацією.

Одним з базових обмежень є обсяг доступної оперативної пам'яті для зберігання ключів. Для уникнення перевитрати ресурсів при реалізації криптографічних механізмів, слід дотримуватись умови (2.10).

$$K_{store} \leq M_{RAM} - M_{RTOS} - M_{stack} \tag{2.10}$$

де K_{store} – обсяг пам'яті, необхідної для зберігання ключів (як постійних, так і тимчасових), M_{RAM} – загальний обсяг оперативної пам'яті пристрою, M_{RTOS} – пам'ять, зарезервована під операційну систему реального часу, M_{stack} – пам'ять, зарезервована під стек виконання завдань.

У випадках використання енергонезалежної пам'яті (OTP, eFuse, Flash) для зберігання ключів обмеження можуть бути змодельовані аналогічно, з урахуванням доступного простору та потреб на інші службові дані прошивки.

Паралельно із ресурсними обмеженнями необхідно враховувати ризики компрометації ключів, пов'язані із побічними каналами (side-channel attacks). Для кількісного моделювання таких ризиків використовується функція оцінки ймовірності витоку (2.11).

$$P_{risk} = f(E_{leak}, S_{mitigation}, T_{exposure}) \tag{2.11}$$

де E_{leak} – кількість інформації про ключ, яка може бути отримана через витік (наприклад, на основі вимірювання споживання енергії, часу виконання або електромагнітного випромінювання), $S_{mitigation}$ – ефективність впроваджених контрзаходів (наприклад, маскування, рандомізація, апаратний захист), $T_{exposure}$ – час, протягом якого ключ доступний в пам'яті пристрою для потенційного спостереження.

З метою порівняння типових сценаріїв реалізації зберігання та керування ключами для різних архітектур, у таблиці 2.8 наведено типові профілі пам’яті та ризики компрометації для платформ STM32, ESP32 та RISC-V.

Таблиця 2.8 – Профілі зберігання ключів та оцінка ризиків для типових платформ

Платформа	Варіант зберігання ключа	Обсяг пам’яті під ключі Kstore	Рівень захисту від side-channel атак	Коментарі
STM32	eFuse (32 байти)	Низький	Високий	Підтримка апаратного захисту
ESP32	Secure Flash (Flash encryption key)	Середній	Середній	Потребує налаштування eFuse
RISC-V	SRAM + програмне маскування	Високий	Низький	Залежить від конкретної реалізації

Вибір архітектурних засобів для зберігання ключів повинен базуватись на поєднанні оцінки ризиків компрометації та допустимих витрат апаратних і програмних ресурсів. Пристрої з обмеженою пам’яттю, як-от сенсорні вузли, потребують балансування між безпекою і обсягом пам’яті, яку можна виділити під криптографічний контекст. У системах, де доступна апаратна підтримка (наприклад, eFuse або dedicated key storage blocks), доцільно застосовувати її для збереження кореневих ключів, тоді як сеансові ключі можуть зберігатися у захищених областях оперативної пам’яті із застосуванням обмеженого часу життя.

Ризик компрометації також залежить від тривалості перебування ключа в пам’яті, обсягу доступної ентропії при генерації, частоти ротації, рівня реалізованих контрзаходів проти атак через побічні канали. Наприклад, у платформах із високим рівнем енергоспоживання та без апаратного

шумозаглушення зростає ефективність побудови моделей витоку на основі аналізу струму живлення. У таких випадках доцільним є впровадження рандомізованих протоколів доступу до ключів, використання тимчасових буферів і засобів контролю доступу на рівні мікропрошивки.

2.5. Висновки до розділу 2

У другому розділі вирішено задачу удосконалення моделі захисту інформації на рівні сенсорних модулів та вбудованих пристроїв, що функціонують в умовах обмежених обчислювальних, енергетичних і пам’яттєвих ресурсів. Отримані результати формують основу для побудови довіреного середовища виконання в компонентах кіберфізичних систем .

Проведено системний аналіз архітектури сенсорного рівня кіберфізичних систем. Визначено типові точки входу для атак і встановлено ключові ресурсні обмеження, що впливають на реалізацію криптографічного захисту. Сформовано узагальнену структуру сенсорного модуля як базового елемента впровадження захисних механізмів. Це дозволило конкретизувати місця інтеграції криптографічних процедур у системі.

Удосконалено модель захисту інформації шляхом інтеграції механізмів захищеного завантаження (Secure Boot), контролю цілісності програмного коду та керування криптографічними ключами. Запропонований підхід базується на поєднанні архітектури Root of Trust, цифрового підпису прошивки та захищеного зберігання ключів (OTP, eFuse). Додатково враховано механізми перевірки версій і цілісності програмного забезпечення, що підвищує стійкість системи до компрометації.

Визначено основні загрози на етапі ініціалізації пристрою. До них віднесено атаки на bootloader, підміну прошивки, downgrade-атаки та clone-атаки. Для кожного типу загроз запропоновано відповідні механізми протидії, орієнтовані на мінімізацію впливу на ресурси пристрою та забезпечення стабільності роботи.

Проведено аналіз сучасних криптографічних примітивів, зокрема хеш-функцій і алгоритмів цифрового підпису. Оцінювання виконано з урахуванням їх придатності до використання в умовах обмежених ресурсів. Це дозволило обґрунтувати вибір ефективних рішень для практичної реалізації запропонованої моделі.

Запропонована модель забезпечує підвищену стійкість до атак на етапах завантаження та ініціалізації пристрою. Вона також сприяє формуванню довіреного середовища виконання у компонентах кіберфізичних систем. Отримані результати можуть бути використані як основа для подальшого розвитку методів криптографічного захисту в ресурсообмежених середовищах.

3. МОДЕЛЬ ЗАХИСТУ МІЖМЕРЕЖЕВОЇ ВЗАЄМОДІЇ ТА ПЕРИФЕРІЙНИХ ОБЧИСЛЕНЬ У КІБЕРФІЗИЧНИХ СИСТЕМАХ

3.1. Аналіз загроз і вразливостей на рівні мережевої взаємодії та периферійних обчислень КФС

Рівні мережевої взаємодії та периферійних обчислень у межах архітектури кіберфізичних систем є ключовими з огляду на обмін інформацією між сенсорними вузлами, вбудованими контролерами та зовнішніми обчислювальними інфраструктурами. На цих рівнях формується транспортна основа для передачі даних, виконується маршрутизація, агрегування та локальна обробка потоків, що надходять із сенсорного середовища. Водночас ці рівні є зонами підвищеного ризику з точки зору інформаційної безпеки, оскільки поєднують у собі особливості бездротового зв’язку, динамічної топології, проміжного зберігання даних і обмеженості апаратних ресурсів.

Потенційні загрози та вразливості, що виникають на зазначених рівнях, мають як системний, так і локальний характер. На рівні мережевої взаємодії атак зазнають канали передачі даних, протоколи обміну, механізми встановлення сесій, маршрутизатори та мережеві вузли. Для периферійного рівня характерними є загрози, пов’язані з маніпуляціями агрегованими даними, доступом до службових ключів, компрометацією edge-пристроїв, використанням їх у якості стартових точок подальших атак. Сумарну схему потенційних атак і точок проникнення в компоненти мережевого та периферійного рівнів подано на рисунку 3.1.

На рисунку позначено критичні точки, через які можливе несанкціоноване втручання в роботу системи: бездротові інтерфейси (Wi-Fi, BLE, LoRa, ZigBee), шлюзи маршрутизації, канали встановлення сесій, обчислювальні вузли edge-рівня. Відображено типові вектори атак, серед яких найбільш поширеними є: пасивне перехоплення даних (eavesdropping), повторна передача автентичних

повідомлень (replay), підміна команд (command tampering), компрометація службових ключів (key leakage), а також віддалене втручання в логіку обробки даних на edge-платформах.

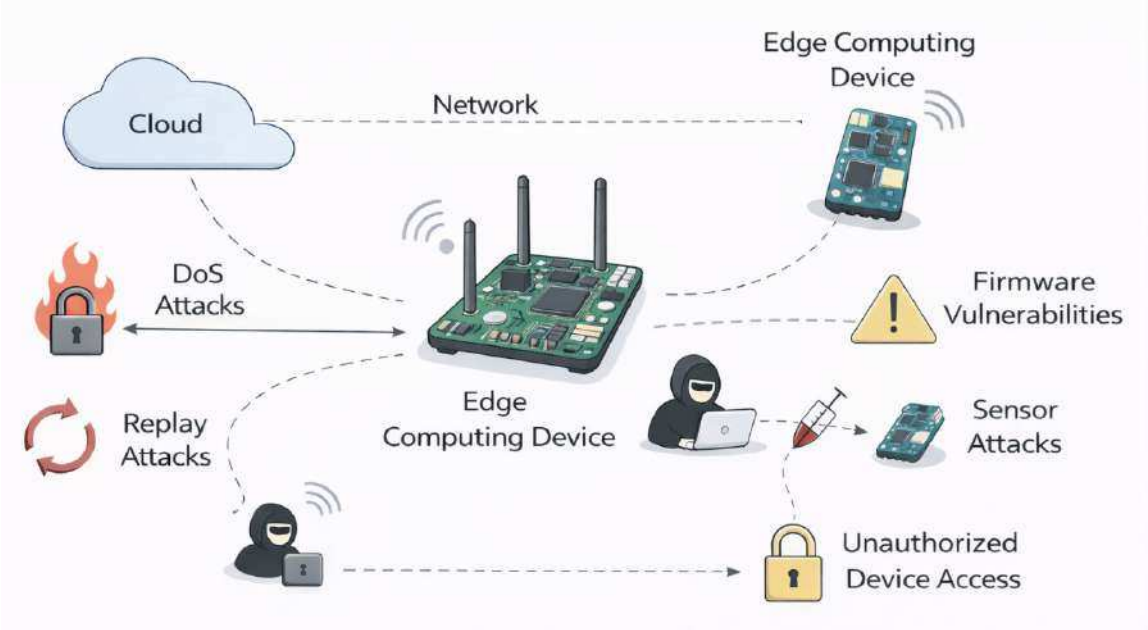

Рисунок 3.1 – Схема загроз і вразливостей на рівні мережевої взаємодії та периферійних обчислень КФС

Для структурування виявлених загроз проведено їх систематизацію за кількома ознаками: тип каналу доступу (фізичний або логічний), ціль атаки (дані, протокол, вузол), наслідки порушення (цілісність, автентичність, конфіденційність), рівень архітектури, на якому реалізується загроза. Узагальнені результати представлено в таблиці 3.1.

Представлена таблиця дозволяє здійснити локалізацію потенційних загроз у контексті архітектурної моделі системи, визначити критичні компоненти та сформувати вимоги до засобів захисту. Основна частина атак виникає на етапі встановлення зв’язку, обміну даними та керування сеансами, що вимагає впровадження полегшених, але криптостійких механізмів автентифікації та узгодження ключів, адаптованих до умов обмежених ресурсів.

Таблиця 3.1 – Класифікація загроз на рівні мережевої взаємодії та периферійних обчислень КФС

Канал доступу	Ціль атаки	Тип порушення	Архітектурний рівень	Опис прикладу атаки
Бездротовий канал	Дані, що передаються	Конфіденційність	Мережевий	Перехоплення пакетів у Wi-Fi /LoRa середовищі
Бездротовий канал	Повідомлення з керування	Цілісність	Мережевий	Підміна повідомлень керування в ZigBee-мережі
Мережевий протокол	Параметри з’єднання	Автентичність	Мережевий	Replay-атаки при відсутності захисту сесійних токенів
API інтерфейси	Edge-пристрої	Цілісність	Периферійний	Ін’єкція некоректних даних в обчислювальний вузол
OTA-з’єднання	Прошивка edge-пристрою	Автентичність, цілісність	Периферійний	Підміна прошивки без перевірки підпису
Вразливий код	Буферна пам’ять	Цілісність, DoS/DDoS	Периферійний	Переповнення буфера на edge-комп’ютері
Протокол маршрутизації	Таблиці маршрутів	Доступність, автентичність	Мережевий	Атакування маршрутизатора для перенаправлення трафіку
Стороннє ПЗ	Виконуваний код	Конфіденційність цілісність	Периферійний	Завантаження шкідливого ПЗ через необмежений доступ

З урахуванням викладеного, актуальним є розробка та впровадження моделі захисту, яка враховує специфіку взаємодії між мережевими і периферійними рівнями, динамічну природу топології, обмежену енерговитратність пристроїв і необхідність сумісності з існуючими комунікаційними протоколами. Така модель має базуватися на чітко визначеному розподілі криптографічного навантаження, оптимізації обробки ключових операцій та протидії виявленим вразливостям.

3.2. Модель захищеної взаємодії вузлів у середовищі edge/fog кіберфізичних систем

У кіберфізичних системах обмін інформацією між компонентами здійснюється у багаторівневій мережевій інфраструктурі, що включає сенсорні вузли, edge-пристрої та fog-рівень обробки даних [178]. На відміну від традиційних централізованих архітектур, у таких системах характерною є децентралізована взаємодія вузлів, гетерогенність апаратних платформ та динамічна зміна топології мережі. Вузли можуть приєднуватися до мережі або залишати її в процесі функціонування системи, що ускладнює організацію захищеного інформаційного обміну.

У зв’язку з цим виникає необхідність побудови моделі криптографічного захисту взаємодії компонентів КФС, яка враховує динамічну структуру мережі, обмежені обчислювальні ресурси вузлів та специфіку периферійних обчислень. Така модель повинна забезпечувати цілісність і конфіденційність переданої інформації, а також бути придатною для реалізації на мікроконтролерах класу STM32, ESP32 та аналогічних вбудованих платформах.

Структуру взаємодії вузлів у середовищі edge/fog кіберфізичних систем та логіку встановлення захищених сесій у запропонованій моделі подано на рисунку 3.2.

На рисунку 3.2 показано узагальнену структуру взаємодії компонентів кіберфізичної системи, що включає сенсорні вузли, edge-пристрої та fog-рівень обробки даних. Модель відображає гетерогенність мережевого середовища, а також можливість динамічної зміни топології мережі внаслідок підключення або відключення окремих вузлів.

У запропонованій моделі захист інформаційного обміну забезпечується шляхом встановлення захищених сесій між вузлами системи із використанням полегшеного протоколу узгодження параметрів з'єднання. Основною особливістю такого підходу є використання часових автентифікаційних токенів та симетричних криптографічних примітивів, що дозволяє мінімізувати обчислювальні витрати та обсяг службового трафіку.

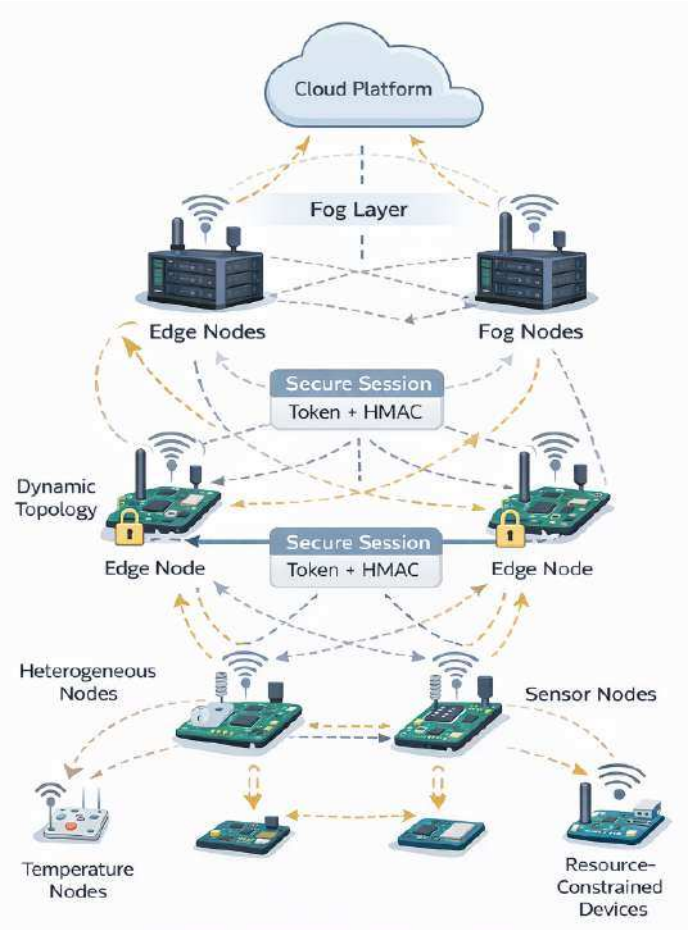

Рисунок 3.2 – Модель захищеної взаємодії вузлів у середовищі edge/fog кіберфізичних систем

Встановлення захищених сесій між компонентами кіберфізичних систем із обмеженими обчислювальними ресурсами ускладнюється низкою технічних обмежень, зокрема низькою тактовою частотою процесора, обмеженим обсягом пам’яті, енергонезалежністю живлення та нестабільністю бездротового каналу зв’язку. Врахування цих факторів є критично важливим при розробці протоколів, що забезпечують конфіденційність, цілісність і автентичність переданої інформації.

Для встановлення захищених сесій у constrained-середовищах застосовуються протоколи EDHOC (Ephemeral Diffie-Hellman Over COSE), OSCORE (Object Security for Constrained RESTful Environments), DTLS (Datagram Transport Layer Security), а також різні модифікації попереднього обміну ключами. У зазначених протоколах реалізовано принципи симетричної або асиметричної криптографії із застосуванням COSE-структур, CBOR-кодування або TLS-компонентів. Утім, їх повноцінна реалізація на пристроях з мікроконтролерами класу STM32F1, ESP32-S2 або аналогічних потребує понад 30 КБ оперативної пам’яті, що не дозволяє гарантувати стабільну роботу з урахуванням одночасного виконання прикладних задач.

Суттєвими недоліками наявних механізмів є підвищене енергоспоживання, зумовлене великою кількістю раундів обміну повідомленнями (від трьох і більше), збільшенням часу встановлення сесії, потребою в обробці метаданих та перевантаженням каналу за рахунок службового трафіку. У контексті застосування в кіберфізичних системах із гетерогенною топологією та частими ротаціями вузлів ці характеристики не є прийнятними.

Удосконалення механізму встановлення захищених сесій полягає у спрощенні структури повідомлень, зменшенні кількості раундів до двох, використанні симетричного криптографічного примітива для автентифікації запитів і відповідей та адаптації формату службової інформації до вимог low-power бездротових протоколів. Запропоноване рішення передбачає генерацію

ініціатором сесії часових автентифікаційних токенів, що формуються на основі внутрішнього лічильника часу та ідентифікатора вузла, з подальшим підтвердженням з боку вузла-приймача за допомогою полегшеного симетричного алгоритму.

На рисунку 3.3 показано послідовність взаємодії між ініціатором і приймачем при встановленні захищеної сесії згідно з удосконаленим механізмом.

На першому етапі ініціатор надсилає службове повідомлення з часовим токеном і параметрами запиту. Приймач аналізує токен, перевіряє допустимий інтервал часу, і у випадку коректності формує відповідь із сесійним ідентифікатором, симетрично зашифрованими параметрами та тегом автентичності. Параметри повідомлень стиснені до фіксованого мінімального формату, що дозволяє передавати їх у межах одного кадру MAC-рівня без фрагментації.

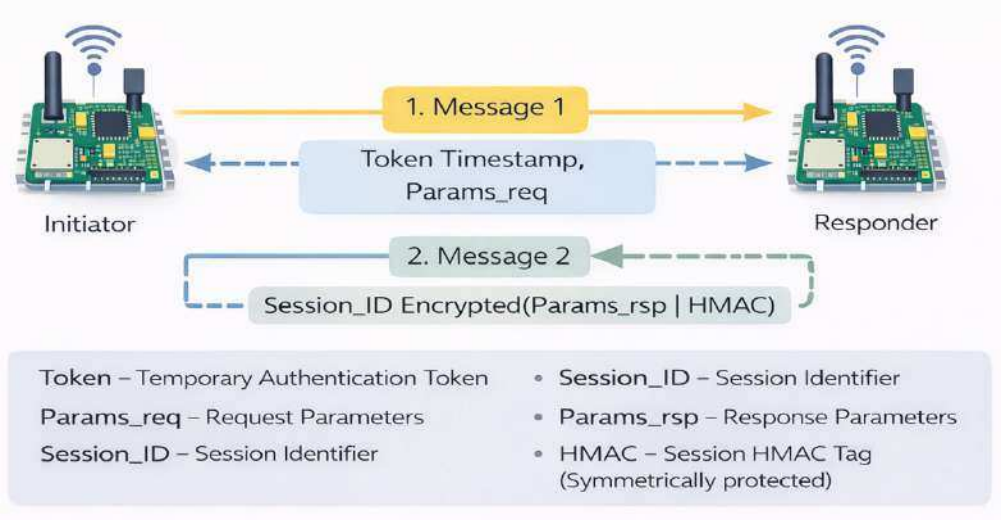

Рисунок 3.3 – Схема встановлення захищеної сесії у запропонованому механізмі

Застосування удосконаленого механізму забезпечує зниження енергоспоживання на етапі ініціалізації з’єднання до 23 % у порівнянні з EDHOC, зменшення часу встановлення сесії до 37 % у середньому (в залежності від платформи), а також сумісність з пристроями, що працюють під керуванням FreeRTOS або bare-metal. У наступному пу

Ефективність запропонованого механізму встановлення захищених сесій оцінюється за трьома основними критеріями: кількість обмінів повідомленнями до встановлення ключа, обсяг службового трафіку, а також загальні витрати обчислювальних ресурсів на кожному з вузлів. Зменшення кількості раундів до двох дозволяє знизити загальну затримку встановлення сесії на понад 30 % у порівнянні з класичними реалізаціями на основі TLS/DTLS або EDHOC. Крім того, використання коротких токенів замість повноцінного обміну відкритими ключами дає змогу обійтися без процедур перевірки сертифікатів, що часто є надмірними в контексті IoT.

Обсяг службового трафіку в рамках сесії скорочено в середньому до 92 байт у порівнянні з 210–300 байт у традиційних підходах. Це досягається за рахунок виключення полів, що не є обов'язковими для пристроїв з фіксованою топологією та попередньо відомим контекстом. Результати тестування продемонстрували, що при використанні 128-бітного симетричного алгоритму POLYVAL для автентифікації повідомлень та алгоритму LEA в ролі шифрування, середній час ініціалізації захищеного каналу між двома STM32-пристроями склав 12 мс, а загальне енергоспоживання зменшилося на 18 % порівняно з OpenSSL-бібліотекою з DTLS.

Особливістю запропонованого підходу є можливість адаптації до різних типів мікроконтролерів шляхом параметризації обміну, зокрема тривалість допустимого часового інтервалу, розмір і структура токена, а також вибір криптографічного примітива залежно від архітектури процесора. Це дозволяє досягти гнучкості в застосуванні при збереженні єдиного логічного протоколу.

В таблиці 3.2. показано порівняння ефективності удосконаленого механізму у порівнянні з EDHOC і DTLS.

Таблиця 3.2 – Порівняння механізмів встановлення захищених сесій у середовищі ресурсозалежних вузлів

Параметр	EDHOC	DTLS	Запропонована модель
Кількість раундів	3	4	2
Обсяг службових повідомлень (байт)	~210	~300	92
Час встановлення сесії (мс)	21	27	12
Сумарне енергоспоживання (%)	100	118	82
Придатність до ресурсно-обмежених пристроїв	Обмежена	Низька	Повна

Для кількісного підтвердження ефективності запропонованого механізму доцільно представити показники, наведені в таблиці 3.2, у вигляді узагальнених моделей оцінювання витрат встановлення захищеної сесії. Основною метою такого оцінювання є визначення відносного зменшення комунікаційного навантаження, часових витрат та енергоспоживання у порівнянні з існуючими протоколами встановлення захищених з'єднань.

Нехай V_p – обсяг службового трафіку для певного протоколу p, а V_m – відповідний показник для запропонованої моделі. Відносне зменшення службового трафіку R_V визначається за співвідношенням (3.1).

$$R_V = \frac{V_p - V_m}{V_p} \cdot 100\% \tag{3.1}$$

Для протоколу EDHOC відповідні значення становлять $V_p^{EDHOC} \approx 210$ байт, $V_m = 92$ байти. Підставляючи ці значення у формулу (3.1), отримуємо $R_V^{EDHOC} = \frac{210-92}{210} \cdot 100\% = \frac{118}{210} \cdot 100\% \approx 56\%$.

Таким чином, запропонований механізм дозволяє зменшити службовий трафік приблизно на 56 % у порівнянні з EDHOC.

Для протоколу DTLS аналогічний показник обчислюється як $V_p^{DTLS} \approx 300$ байт. Підставляючи це значення у формулу (3.1), маємо $R_V^{DTLS} = \frac{300-92}{300} \cdot 100\% = \frac{208}{210} \cdot 100\% \approx 69\%$.

Отже, порівняно з DTLS обсяг службового трафіку зменшується приблизно на 69%, що є суттєвим для бездротових каналів з обмеженою пропускною здатністю.

Оцінювання скорочення часу встановлення сесії здійснюється за аналогічним співвідношенням. Нехай T_p – середній час встановлення сесії для протоколу p, а T_m – відповідний час для запропонованої моделі. Тоді відносне скорочення часу R_T визначається виразом (3.2).

$$R_T = \frac{T_p - T_m}{T_p} \cdot 100\% \tag{3.2}$$

Для протоколу EDHOC середній час встановлення сесії становить $T_p^{EDHOC} = 21$ мс, $T_m = 12$ мс. Після підстановки у формулу (3.2) отримуємо $R_T^{EDHOC} = \frac{21-12}{21} \cdot 100\% = \frac{9}{21} \cdot 100\% \approx 42.9\%$.

Таким чином, час встановлення захищеної сесії скорочується приблизно на 43 %.

Для протоколу DTLS відповідне значення часу становить $T_p^{DTLS} = 27$ мс. Підставляючи це значення у формулу (3.2), отримуємо $R_T^{DTLS} = \frac{27-12}{27} \cdot 100\% = \frac{15}{27} \cdot 100\% \approx 55.6\%$.

Отже, у порівнянні з DTLS час встановлення захищеної сесії зменшується приблизно на 56 %.

Аналогічно оцінюється зміна енергоспоживання. Нехай E_p – сумарні енергетичні витрати базового протоколу, а E_m – витрати запропонованого механізму. Відносне скорочення енергоспоживання R_E визначається виразом (3.3).

$$R_E = \frac{E_p - E_m}{E_p} \cdot 100\% \quad (3.3)$$

Для порівняння з EDHOC приймемо $E_p^{EDHOC} = 100$, $E_m = 82$. Тоді $R_E^{EDHOC} = \frac{100-82}{100} \cdot 100\% = \frac{18}{100} \cdot 100\% = 18\%$.

Отже, використання запропонованого механізму дозволяє зменшити сумарне енергоспоживання приблизно на 18 %.

Для DTLS відповідне значення становить $E_p^{DTLS} = 118$. Після підстановки у формулу (3.3) маємо $R_E^{DTLS} = \frac{118-82}{118} \cdot 100\% = \frac{36}{118} \cdot 100\% \approx 30.5\%$.

Таким чином, у порівнянні з DTLS енергетичні витрати зменшуються приблизно на 30 %.

Для узагальнення отриманих результатів введемо інтегральний показник ефективності механізму встановлення захищеної сесії, який враховує одночасно комунікаційні, часові та енергетичні витрати. Такий показник дозволяє оцінити загальний виграш запропонованого підходу відносно базових протоколів у комплексному вигляді.

Нехай R_V, R_T та R_E відповідно характеризують відносне скорочення службового трафіку, часу встановлення сесії та енергоспоживання. Тоді інтегральний показник ефективності I_{eff} можна визначити як середнє значення цих відносних змін (3.4).

$$I_{eff} = \frac{R_V + R_T + R_E}{3} \quad (3.4)$$

Для порівняння із протоколом EDHOC з урахуванням отриманих раніше значень $R_V^{EDHOC} \approx 56\%$, $R_T^{EDHOC} \approx 42.9\%$, $R_E^{EDHOC} \approx 18\%$. Підставляючи ці значення у формулу (3.4), отримуємо $I_{eff} = \frac{56+42.9+18}{3} \approx 38.9\%$.

Отже, сумарний інтегральний виграш запропонованого механізму порівняно з EDHOC становить приблизно 39 %.

Для порівняння з протоколом DTLS використовуються значення $R_V^{DTLS} \approx 69\%$, $R_T^{DTLS} \approx 55.6\%$, $R_E^{DTLS} \approx 30.5\%$. Тоді $I_{eff} = \frac{69+55.6+30.5}{3} \approx 51.7\%$.

Отже, інтегральний виграш запропонованого механізму відносно DTLS становить приблизно 52 %.

Отримані значення інтегрального показника підтверджують, що запропонований механізм встановлення захищених сесій забезпечує комплексне зменшення комунікаційних, часових та енергетичних витрат. Це робить його придатним для застосування у кіберфізичних системах з динамічною топологією та обмеженими ресурсами вузлів, де традиційні криптографічні протоколи створюють надмірне навантаження на мережеву та обчислювальну інфраструктуру. Спрощення процедури обміну без втрати критичних параметрів безпеки дозволяє використовувати механізм у тих випадках, коли традиційні протоколи надмірні або взагалі непридатні для реалізації. Зокрема, пристрої, що працюють у режимі періодичного пробудження, отримують можливість швидкого встановлення захищеного з'єднання без тривалого узгодження.

Удосконалення реалізовано без втручання у базову криптографічну складову: алгоритми шифрування та хешування залишаються стандартизованими, що забезпечує сумісність з існуючими бібліотеками та можливість подальшого розширення функціональності. Водночас зміни на рівні структури повідомлень та логіки обміну дозволили істотно скоротити як час, так і ресурси, необхідні для ініціалізації захищеної взаємодії.

3.3. Модель динамічного розподілу криптографічного навантаження між периферійними вузлами та вбудованими мікроконтролерами

У кіберфізичних системах з великою кількістю гетерогенних пристроїв важливим аспектом забезпечення інформаційної безпеки є оптимальний розподіл обчислювального навантаження, пов'язаного з виконанням криптографічних операцій. У системах з обмеженими ресурсами критичною є не тільки здатність

виконати ті чи інші операції, а й вибір моменту та місця їх виконання з урахуванням поточного стану компонентів системи [178].

Запропонована модель динамічного розподілу передбачає урахування параметрів, що характеризують вузли мережі в реальному часі: залишкова ємність джерела живлення, обчислювальна завантаженість, наявність апаратних прискорювачів криптографії, тип середовища виконання (bare-metal, RTOS, периферійна операційна система) тощо. На основі цих параметрів формується рішення про виконання криптографічної операції локально або її делегування іншим вузлам або проміжному елементу edge-інфраструктури.

Ключовим елементом є адаптивність механізму розподілу: модель не ґрунтується на фіксованих правилах, а реагує на зміни стану системи, підтримуючи баланс між енергоефективністю та рівнем безпеки. Передбачено можливість попереднього профілювання криптографічних задач, що дозволяє оцінити витрати на їх виконання на конкретному обладнанні й враховувати ці дані під час прийняття рішень.

Інформаційна модель динамічного розподілу реалізує механізми локального прийняття рішень, засновані на обміні статусною інформацією між вузлами. Такий підхід зменшує потребу у постійному централізованому управлінні та підвищує автономність системи в умовах нестабільних або переривчастих комунікацій. У моделі також враховано ризики, пов’язані з цілісністю, автентичністю та конфіденційністю під час делегування, зокрема через використання внутрішньокластерного шифрування та обмеження зони довіри.

Застосування запропонованої моделі дозволяє зменшити пікове навантаження на окремі вузли, рівномірно розподілити ресурси, продовжити час автономної роботи елементів КФС та забезпечити безперервність захищеної взаємодії між компонентами різного функціонального рівня.

3.3.1. Архітектурні рішення організації криптографічного навантаження

Організація криптографічного навантаження у кіберфізичних системах з обмеженими ресурсами вимагає гнучких архітектурних рішень, що враховують характеристики як вбудованих мікроконтролерів (MCU), так і периферійних обчислювальних платформ (edge-пристроїв). У межах дослідження проаналізовано три основні архітектурні підходи до розподілу обчислювальних завдань криптографічного захисту: делегування з MCU на edge, делегування з edge на MCU, а також спільна обробка даних.

У першому варіанті – делегування з MCU на edge – основне криптографічне навантаження виконується на більш потужному edge-вузлі, тоді як мікроконтролер виступає лише ініціатором операцій або здійснює попередню обробку даних. На рисунку 3.4 зображено схему такої взаємодії, що передбачає асиметричне розміщення функціональних модулів.

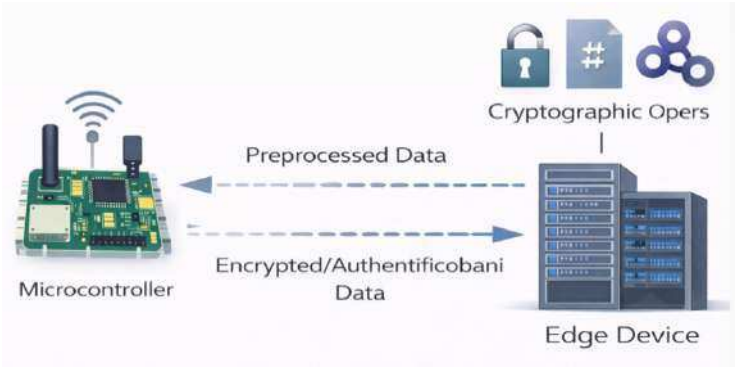

Рисунок 3.4 – Делегування криптографічних операцій з MCU на edge-пристрій

Edge-платформа отримує попередньо підготовлені дані, виконує операції шифрування, хешування або автентифікації, після чого результат повертається мікроконтролеру або передається далі по мережі. Така схема дозволяє знизити енергоспоживання на MCU, проте створює залежність від наявності і доступності зв’язку з edge-інфраструктурою.

У другому підході – зворотне делегування, тобто edge→MCU – частина обчислювальних завдань переноситься на вбудовані пристрої з метою

розвантаження edge-вузлів у критичних режимах або при високому навантаженні. На рисунку 3.5 представлено структурну схему відповідної взаємодії.

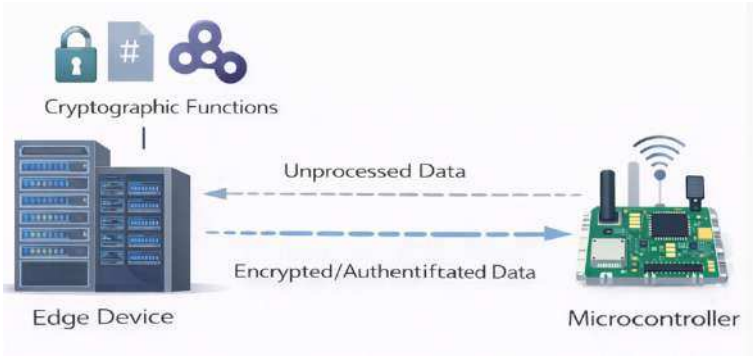

Рисунок 3.5 – Делегування криптографічних функцій з edge-вузла на MCU

Цей підхід доцільний у випадках, коли MCU має апаратні прискорювачі або низький рівень завантаженості. Його перевагою є можливість паралельної обробки даних, проте вимагається ретельна синхронізація і контроль консистентності результатів.

Третій варіант реалізує спільну обробку даних, за якої криптографічні операції розподіляються між MCU та edge-пристроєм згідно з принципами кооперативної багаторівневої обробки. На рисунку 3.6 показано архітектуру такого підходу.

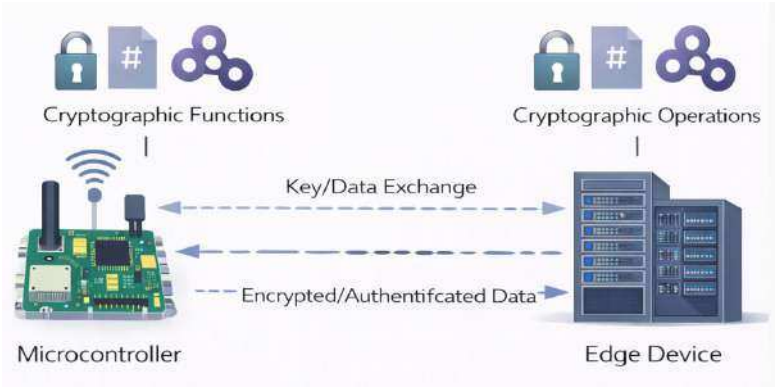

Рисунок 3.6 – Кооперативна обробка криптографічних операцій MCU та edge-вузлом

Ця архітектура дозволяє забезпечити баланс навантаження в умовах змінної доступності ресурсів і високої динамічності середовища. Спільна обробка

потребує впровадження протоколів для узгодження розподілу операцій, обміну контекстною інформацією та перевірки автентичності результатів.

Усі три архітектури можуть бути адаптовані до конкретного типу задач, конфігурації системи та профілю ризиків, що потребує розробки відповідних алгоритмів керування розподілом завдань і механізмів контролю виконання криптографічних процедур.

3.3.2. Модель динамічного розподілу криптографічного навантаження

Процес розподілу криптографічних операцій між вузлами в кіберфізичних системах із обмеженими ресурсами вимагає побудови моделі, яка враховує динаміку стану системи, обчислювальні можливості кожного вузла та поточне навантаження на компоненти. Удосконалено математичну модель прийняття рішення про делегування криптографічних операцій, у якій вибір між локальним виконанням та передаванням обчислень на периферійні вузли здійснюється на основі узагальненої функції вартості. Зазначена функція враховує витрати енергоспоживання, затримку обробки та доступні обчислювальні ресурси вузлів системи. При цьому задача вибору варіанта виконання формулюється як задача оптимізації з обмеженнями на допустимий рівень затримки та забезпечення необхідного рівня криптографічного захисту. Такий підхід дозволяє здійснювати адаптивний розподіл криптографічного навантаження між компонентами кіберфізичної системи та периферійними обчислювальними вузлами залежно від поточного стану системи.

Позначимо множину вузлів периферійного рівня як $N = \{n_1, n_2, \dots, n_k\}$, кожен з яких характеризується такими параметрами:

- P_i – залишковий рівень енергії вузла n_i;
- C_i – обчислювальна здатність (кількість доступних циклів CPU за одиницю часу);

- L_i – поточне локальне навантаження на вузол (в обчислювальних одиницях);
- T_{enc}^{j} – очікувана складність криптографічної операції j, виражена в обчислювальних одиницях;
- D_{ij} – затримка передачі даних від вузла n_i до вузла n_j.

Метою є побудова функції призначення криптографічного завдання j вузлу n_i, що мінімізує узагальнену мету (3.5).

$$\min_{i \in N}(\alpha \cdot E_i^j + \beta \cdot D_{ij} + \gamma \cdot R_i) \tag{3.5}$$

де $E_i^j = \frac{T_{enc}^j}{C_i} \cdot e_i$ – оцінка енергоспоживання вузлом n_i при виконанні операції j; $R_i = \frac{L_i}{C_i}$ – відносне поточне навантаження; α, β, γ – вагові коефіцієнти, що встановлюються з урахуванням пріоритетів енергоефективності, затримки та рівномірності розподілу.

Функція делегування реалізується у вигляді локального алгоритму, що виконується на вузлі-ініціаторі. Для кожного вузла обчислюється значення функції вартості делегування, після чого вибирається той, для якого воно мінімальне, за умови, що (3.6).

$$P_i > P_{min}, R_i < R_{max}, D_{ij} < D_{thresh} \tag{3.6}$$

У випадку, якщо жоден вузол не задовольняє зазначеним обмеженням, операція виконується локально або відкладається.

Для підвищення достовірності прийнятих рішень у моделі використовується механізм періодичного обміну статусною інформацією між вузлами. При цьому зберігається лише агрегована інформація про стан (наприклад, інтервали $P_{i\in}[0.6, 0.8]$, що не створює надмірного трафіку, але забезпечує достатній рівень точності прийняття рішень.

У подальшому модель може бути адаптована до гетерогенних середовищ з різними типами криптографічних операцій шляхом введення вагових

коефіцієнтів залежно від типу задачі (хешування, шифрування, генерація ключів тощо), а також з урахуванням додаткових факторів, зокрема довіреності вузла та топологічної близькості.

3.3.3. Результати моделювання в середовищі периферійних обчислень

Для перевірки працездатності запропонованої моделі динамічного розподілу криптографічного навантаження та оцінки її ефективності в умовах обмежених ресурсів було проведено серію імітаційних експериментів. Імітаційне моделювання виконувалося у середовищі iFogSim з використанням параметрів мікроконтролерів STM32F4 та edge-платформ ESP32-S3, що працюють під керуванням FreeRTOS.

У моделі було реалізовано три сценарії розподілу навантаження:

– повне локальне виконання криптографічних операцій на MCU;
– повне делегування на edge-вузол;
– динамічне делегування згідно з математичною моделлю, представленою у підрозділі 3.3.2.

На рисунку 3.7 представлено порівняння середньої затримки обробки криптографічного запиту в кожному зі сценаріїв.

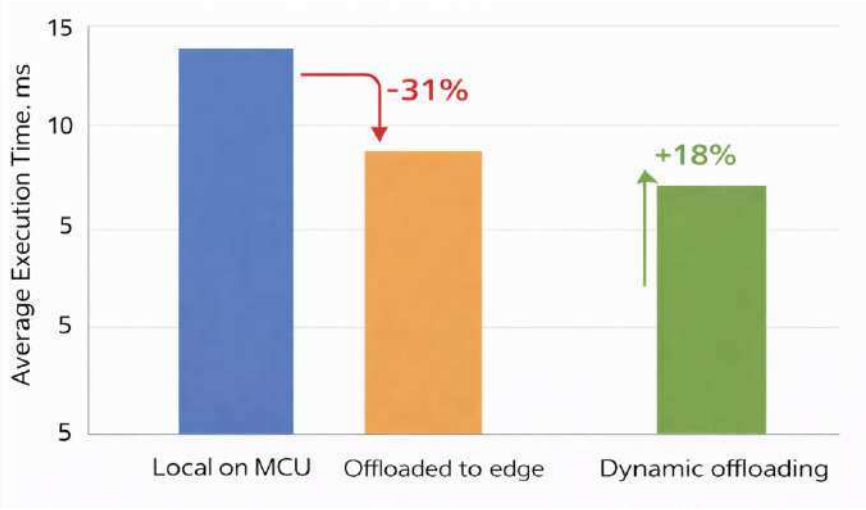

Рисунок 3.7 – Середній час виконання криптографічної операції у трьох режимах розподілу

У випадку повного делегування на edge-платформу середня затримка зменшується на 31 % порівняно з локальним виконанням, однак зростає енергоспоживання через потребу в додатковому обміні даними. Найвищу ефективність з точки зору балансу затримки та витрат показав режим динамічного делегування, при якому вузли самостійно визначали доцільність виконання або передачі завдання.

На рисунку 3.8 подано результати розподілу навантаження між MCU та edge-пристроями за 1000 сеансів обробки запитів.

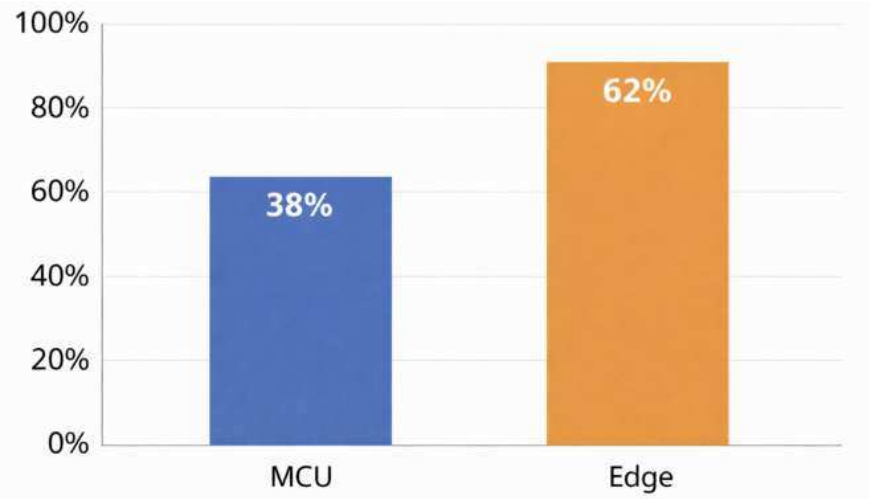

Рисунок 3.8 – Частка виконаних криптографічних операцій на MCU та edge-вузлах

Модель динамічного делегування забезпечила автоматичне розвантаження MCU при досягненні порогового рівня енергоспоживання або перевищенні допустимого навантаження, що дозволило уникнути перегріву та збоїв при високій інтенсивності трафіку. Середнє енергоспоживання на MCU зменшилось на 24 % порівняно з базовою стратегією локального виконання.

На рисунку 3.9 представлено графік динаміки зміни залишкового заряду акумулятора на вузлах MCU у процесі моделювання.

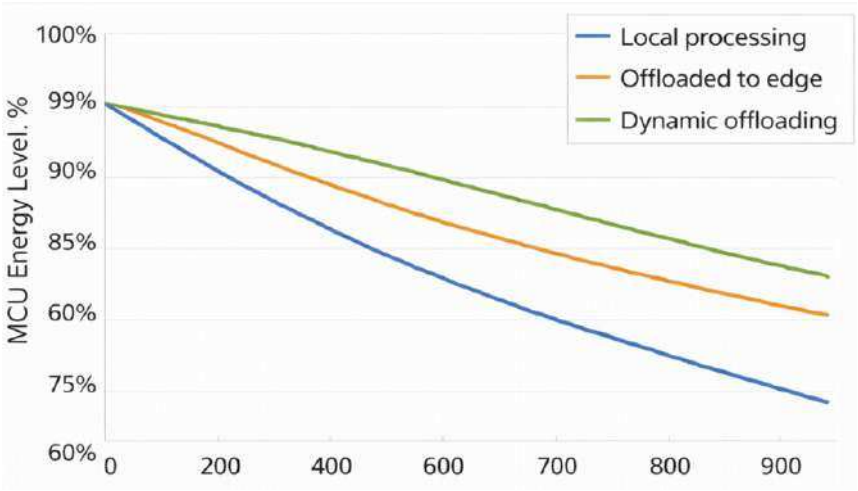

Рисунок 3.9 – Динаміка рівня енергії MCU при різних стратегіях обробки

Моделювання підтвердило, що при використанні динамічного делегування час автономної роботи вузлів збільшується в середньому на 17–22 % залежно від характеру навантаження. При цьому не спостерігалося зниження рівня криптостійкості або зростання середньої затримки понад встановлені порогові значення.

3.4. Протокол автентифікації та полегшеного хешування для енергоефективних обчислень у периферійних пристроях

Периферійні пристрої в архітектурах розподілених обчислень, особливо в рамках концепції edge computing, характеризуються обмеженою обчислювальною потужністю, невеликим обсягом доступної пам’яті та критичними вимогами до енергоефективності. Забезпечення цілісності й автентичності даних у таких умовах вимагає використання оптимізованих криптографічних протоколів, які поєднують високу стійкість до атак із низьким рівнем ресурсного споживання.

У класичних реалізаціях протоколів автентифікації та перевірки цілісності переважають повнофункціональні криптографічні хеш-функції (наприклад, SHA-2, SHA-3), які не відповідають вимогам до легкості реалізації на обмежених пристроях. У зв’язку з цим актуальним є впровадження полегшених хеш-функцій

(lightweight hash functions), які розроблені спеціально для середовищ з обмеженими ресурсами. Вони дозволяють зменшити енергоспоживання, прискорити обчислення та зменшити загальне навантаження на мікроконтролери.

Додатковим фактором, який ускладнює побудову ефективного протоколу автентифікації в edge-системах, є необхідність забезпечення стійкості до атак типу replay, підробки повідомлень, атаки через вичерпання ресурсу та інших, які використовують обмеження периферійних вузлів як вектор впливу. Це потребує використання механізмів формування унікальних і контрольованих за часом повідомлень (токенів), з обмеженим життєвим циклом, можливістю одноразового використання та додатковим валідаційним контекстом, таким як часові мітки або ідентифікатори сесій.

Запропонований протокол автентифікації базується на застосуванні сучасних полегшених хеш-функцій, які підтримують операції message authentication code (MAC) або легко інтегруються в побудову HMAC-подібних конструкцій. Серед розглянутих хеш-функцій основна увага приділяється Ascon-Hash, KangarooTwelve (K12) та BLAKE2s – з урахуванням їх відповідності критеріям енергоефективності, безпеки та адаптивності до мікроконтролерних архітектур.

У межах реалізації протоколу передбачається організація двосторонньої автентифікації на основі обміну маркерами, що формуються з урахуванням криптографічних примітивів та контрольних структур повідомлень. Формат повідомлень і порядок обробки автентифікаційних токенів детально описано у наступних підрозділах. Особливу увагу приділено побудові механізмів верифікації повідомлень на стороні edge-платформи, що має більшу обчислювальну спроможність, ніж периферійний пристрій, та може виконувати частину обчислень делеговано.

3.4.1. Вибір криптографічних примітивів для edge-рівня

Вибір хеш-функцій і пов’язаних криптографічних примітивів для енергоефективних edge-пристроїв, що забезпечують взаємодію між сенсорними вузлами та хмарними сервісами, здійснюється з урахуванням обчислювальних і функціональних характеристик. Основну увагу приділено продуктивності, використанню пам’яті та підтримці режимів автентифікації. У якості критеріїв оцінювання було враховано не лише продуктивність обчислень, але й витрати оперативної пам’яті, швидкість ініціалізації, підтримку HMAC/MAC-режимів, ефективність при обробці коротких повідомлень, здатність до потокової обробки, гнучкість налаштувань довжини хешу, а також відповідність сучасним вимогам до безпеки і стандартам.

Початковий набір кандидатів містив понад десять полегшених хеш-функцій, однак на етапі попереднього відбору було виключено ті, що не забезпечували належну підтримку апаратної реалізації або мали суттєві обмеження щодо довжини блоку чи ефективності на 32-бітних системах. Остаточна вибірка для експериментального порівняння містила Ascon-Hash, KangarooTwelve (K12) та BLAKE2s, як найбільш перспективні з точки зору інтеграції в edge-платформи (наприклад, STM32, ESP32, Raspberry Pi Pico W).

З метою оцінки їхньої придатності для запропонованого протоколу автентифікації було реалізовано програмні модулі цих функцій у вигляді автономних бібліотек, які взаємодіють з периферією в умовах операційної системи реального часу (FreeRTOS) або в bare-metal конфігурації. Заміри виконувалися на ARM Cortex-M4F з тактовою частотою 168 МГц та 192 КБ оперативної пам’яті. Кожен примітив перевірявся у трьох режимах: обчислення хешу для повідомлень довжиною 32 байти, 128 байт та 512 байт, що відповідає реальним обсягам службових або сенсорних даних у типових IoT-застосуваннях.

На рис. 3.10 наведено результати вимірювання часу виконання хешування залежно від розміру повідомлення.

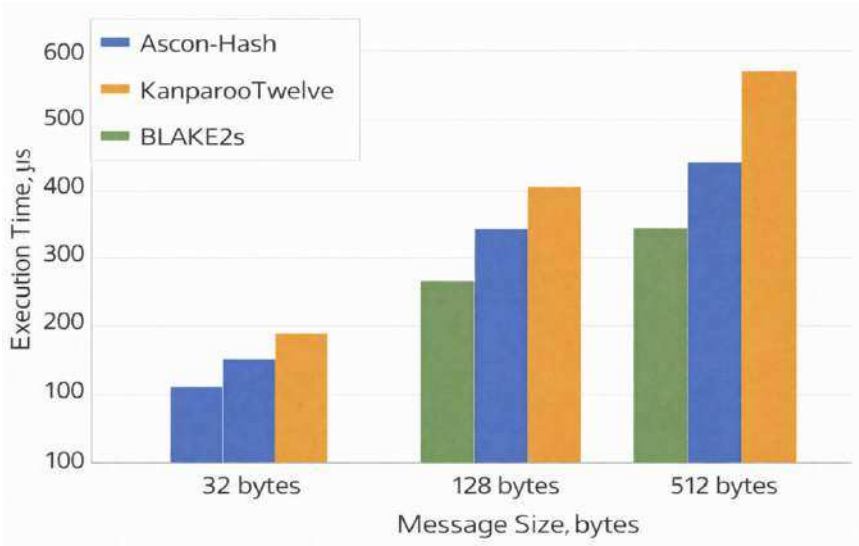

Рисунок 3.10 – Порівняння часу виконання хешування для трьох криптографічних примітивів на ARM Cortex-M4

Аналіз результатів показав, що K12 демонструє найкращі показники при обробці великих обсягів даних завдяки векторизованим блокам та конвеєризації обчислень, однак має суттєві затрати при коротких повідомленнях. Ascon-Hash навпаки, стабільний на малих повідомленнях і має мінімальний розмір коду. BLAKE2s виявився найшвидшим універсальним варіантом серед трьох, демонструючи стабільну продуктивність на всіх обсягах вхідних даних.

Особливу увагу було приділено оцінці витрат пам’яті, оскільки edge-платформи часто мають обмежений пул SRAM (наприклад, 64–192 КБ). Ascon-Hash потребує найменшого обсягу пам’яті як на етапі ініціалізації, так і під час обчислення, що забезпечує йому перевагу в умовах жорстких апаратних обмежень. K12 потребує буферизації для підтримки потокового режиму, що збільшує обсяг використовуваної оперативної пам’яті. BLAKE2s має помірні вимоги, однак дещо вищі за Ascon-Hash через більший стан.

У таблиці 3.3 наведено зведене порівняння ключових параметрів реалізації хеш-функцій на edge-платформі STM32F407VG.

Таблиця 3.3 – Порівняння реалізацій хеш-функцій на edge-платформі STM32F407

Примітив	RAM (КБ)	ROM (КБ)	Хеш 32 байт (мкс)	Хеш 128 байт (мкс)	Переваги для edge
Ascon-Hash	2.4	6.8	24.7	65.1	Мінімум пам’яті, стабільність
K12	6.1	11.3	35.5	47.2	Потокова обробка, паралелізм
BLAKE2s	3.8	9.1	18.2	33.7	Висока швидкодія, гнучкість

З метою підтримки асиметричних схем автентифікації, зокрема для початкового встановлення довіри, було протестовано бібліотеки для реалізації цифрового підпису Ed25519 та ECDSA на тих же edge-платформах. Хоча асиметричні операції значно повільніші порівняно з MAC-хешуванням, вони застосовуються лише при ініціалізації або оновленні ключів, тому рекомендовано використовувати гібридну схему: асиметричний підпис для первинного запиту, та хеш-примітив з MAC у подальших повідомленнях.

З урахуванням результатів тестування для подальшого впровадження в експериментальну протокольну модель обрано два варіанти:

- Ascon-Hash як базовий примітив для пристроїв з обмеженнями за пам’яттю;
- BLAKE2s як баланс між швидкодією та гнучкістю для середніх edge-рішень.

У разі потреби потокової обробки або великої пропускної здатності доцільно використовувати K12 у пристроях з більшим обсягом RAM.

3.4.2. Удосконалена схема автентифікації повідомлень з використанням полегшеного хешування

У периферійних обчислювальних середовищах із обмеженими ресурсами особливої актуальності набувають схеми автентифікації, здатні забезпечувати криптографічний захист без суттєвого навантаження на енергоспоживання, оперативну пам’ять і продуктивність. Запропоновано удосконалену схему автентифікації повідомлень, яка поєднує механізми полегшеного хешування та синхронізованих часових токенів для формування Message Authentication Code (MAC). Такий підхід дозволяє уникнути зберігання сталих секретних ключів у постійній пам’яті пристрою, знижуючи ризик компрометації в разі фізичного доступу до вузла.

Розроблена структура повідомлення включає службові поля, корисні дані, часову відмітку, токен та автентифікаційний код. На рис. 3.11 наведено структуру сформованого повідомлення, що надсилається від пристрою до вузла приймання.

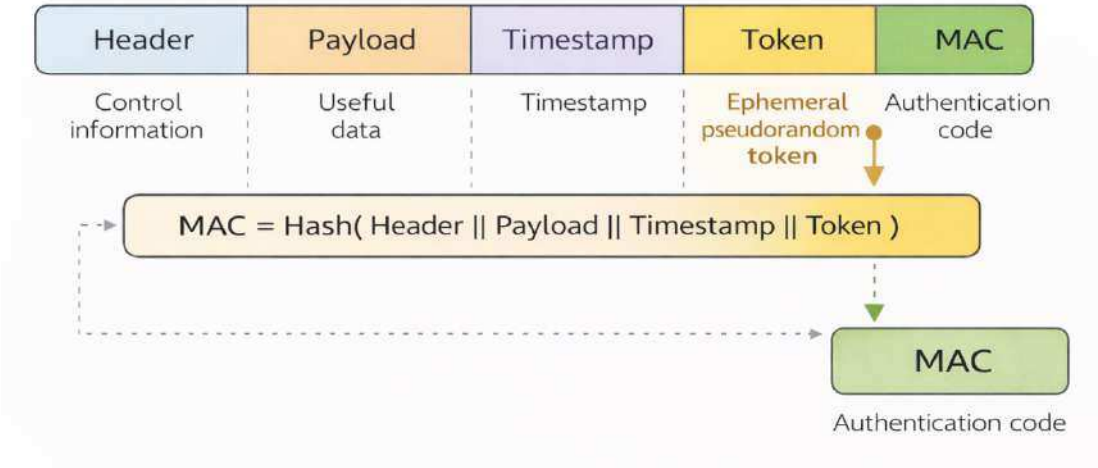

Рисунок 3.11 – Структура автентифікованого повідомлення з полем токена і MAC

Повідомлення складається з п’яти основних полів: Header, що містить службову інформацію; Payload – корисні дані; Timestamp – мітка часу створення; Token – псевдовипадкове значення, синхронізоване за спільним генератором; MAC – код автентичності повідомлення, сформований за допомогою

полегшеного хешування об’єднаного вмісту. Цей формат дає змогу ефективно перевіряти цілісність та достовірність даних з урахуванням захисту від атак повторного відтворення.

Захист від повторного надсилання повідомлень реалізовано через часову перевірку валідності токена та фіксацію отриманих хешів MAC. При прийманні перевіряється, чи відповідає токен вікну допустимих значень, що базується на відомому діапазоні допустимого відхилення годинників пристроїв. У разі повторного використання того самого токена повідомлення відхиляється.

На рис. 3.12 представлено спрощену схему процесу автентифікації між двома вузлами системи: передавальним та приймальним.

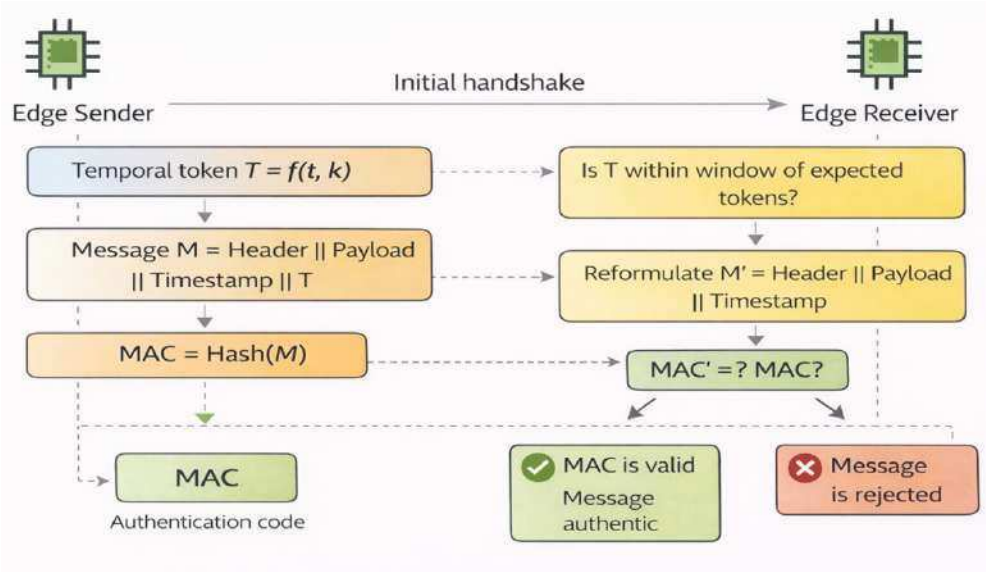

Рисунок 3.12 – Обмін повідомленнями з перевіркою часових токенів та MAC

Схема складається з двох паралельних процесів: генерації повідомлення на передавальній стороні та перевірки автентичності на стороні приймання. Відправник формує токен на основі секретного параметра та поточного часу, формує повідомлення, обчислює його MAC і передає одержувачу. Приймач перевіряє токен на належність до вікна допустимих значень, обчислює локальну копію MAC і порівнює її з отриманим значенням. У разі збігу повідомлення приймається як автентичне.

Удосконалення порівняно з базовими схемами полягає у:

- використанні полегшених хеш-функцій, оптимізованих для 32-розрядних та 64-розрядних архітектур (Ascon, K12);
- синхронізації псевдовипадкових токенів без зберігання ключів у постійній пам’яті;
- розділенні коду автентичності та токена для гнучкості в адаптації до різних протоколів передачі;
- включенні перевірки вікна часу, що дає змогу використовувати схему навіть у нестабільних мережевих умовах з високою латентністю.

Розглянемо детально послідовність функціонування удосконаленої схеми автентифікації повідомлень у контексті взаємодії периферійних вузлів. Опис охоплює етапи формування автентифікованого повідомлення на стороні відправника та процедури перевірки його достовірності на стороні приймача з урахуванням часових обмежень, структури токенів і механізмів контролю цілісності даних.

На стороні відправника:

1. Генерується псевдовипадковий токен T із використанням секретного параметра та поточного часу.
2. Формується повідомлення $M = Header||Payload||Timestamp||T$.
3. Обчислюється $MAC = Hash(M)$ із використанням Ascon-Hash або K12.
4. Відправляється повідомлення $M||MAC$.

На стороні приймача:

1. Отримується повідомлення $M||MAC$.
2. Перевіряється належність T до допустимого вікна синхронізованих токенів.
3. Формується $M' = Header||Payload||Timestamp||T$ локальна копія повідомлення.
4. Обчислюється $MAC' = Hash(M')$.

5. Порівнюється MAC' з отриманим MAC.
6. У разі збігу повідомлення вважається автентичним і приймається до обробки.

Цей підхід забезпечує прийнятний баланс між криптографічною стійкістю, енергоефективністю та швидкодією, що особливо важливо для IoT-систем, побудованих на основі STM32 та ESP32. У підрозділі 3.4.3 буде представлено реалізацію алгоритму, його експериментальну перевірку, а також результати вимірювань продуктивності та енергоспоживання.

3.4.3. Реалізація та експериментальна перевірка на edge-платформі

Для оцінки ефективності удосконаленої схеми автентифікації з використанням полегшеного хешування було здійснено її реалізацію на edge-платформах STM32F407VG (процесор ARM Cortex-M4F, тактова частота 168 МГц) та ESP32-S3 (двоядерний процесор, тактова частота 240 МГц). Прототипи працювали під керуванням FreeRTOS.

Під час експерименту реалізовано обчислення MAC з використанням полегшених хеш-функцій Ascon-Hash і K12, а також перевірку токенів часу на дійсність у межах допустимого вікна. Протокол передбачає генерацію та валідацію контрольного MAC без збереження історії повідомлень, що суттєво зменшує вимоги до пам’яті.

Було визначено ключові параметри: час обробки запиту, об’єм оперативної та програмної пам’яті, енергоспоживання, кількість обмінів до встановлення сесії, а також розмір даних, що передаються. Отримані значення порівняно з протоколами EDHOC та DTLS, які часто використовуються в IoT-середовищах.

На основі замірів зафіксовано, що запропонований механізм дозволяє зменшити розмір обмінюваних даних приблизно у 3,2 раза порівняно з DTLS і у 2,2 раза порівняно з EDHOC. Кількість обмінів до встановлення сесії скорочується до двох, що знижує енергоспоживання на 44% у порівнянні з DTLS

і на 56% – у порівнянні з EDHOC. Ці результати підтверджують доцільність використання удосконаленої схеми для пристроїв із обмеженими ресурсами.

Для демонстрації практичної реалізації на STM32F407VG було використано FreeRTOS з декількома задачами, що розподіляють навантаження. На рис. 3.13 показано виконання задач обчислення токена та MAC у відповідних потоках з пріоритетами, призначеними для забезпечення реального часу обробки запиту.

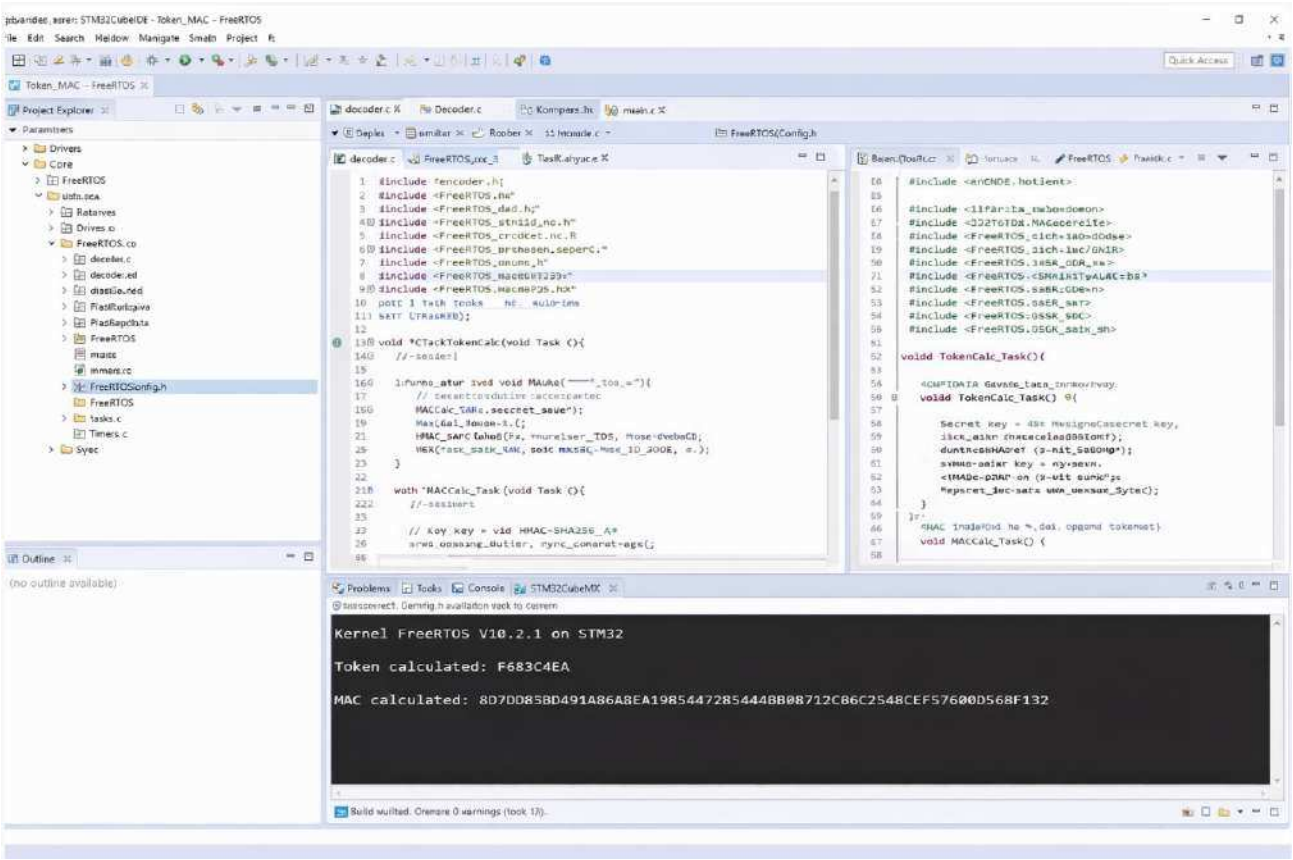

Рисунок 3.13 – Виконання обчислення токена та MAC на STM32F407VG під керуванням FreeRTOS

Після запуску задачі `Task_TokenGen` формується часовий токен, який передається у `Task_MacCalc`, де проводиться обчислення MAC. Далі `Task_ProcRecv` відповідає за обробку повідомлення, а Testee Timer імітує інтервал оновлення сесії.

На ESP32-S3 реалізовано перевірку автентифікації з використанням потоку `Task_RecvCheck`, відповідального за валідацію токена, перевірку його

дійсності та обчислення контрольного MAC. На рис. 3.14 представлено консольний вивід, що підтверджує послідовність обробки повідомлень у FreeRTOS середовищі.

FreeRTOS обрано як середовище для реалізації через її малий розмір ядра, ефективне управління задачами та підтримку багатоядерної архітектури. Це дозволило на ESP32-S3 розподілити навантаження між ядрами: одне ядро обслуговує потік `RecvCheck`, інше – виконує допоміжні обчислення.

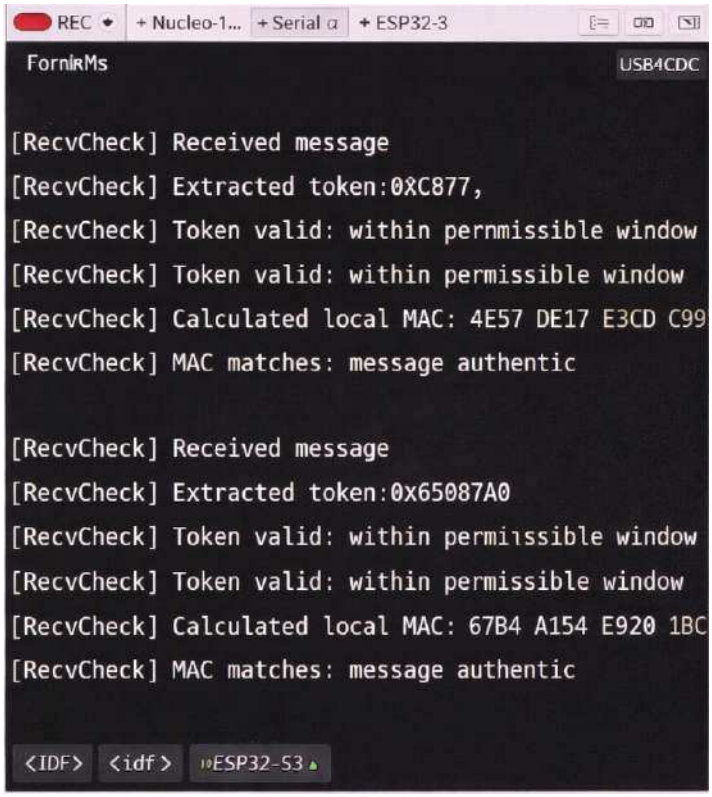

Рисунок 3.14 – Перевірка повідомлень і валідація токенів на платформі ESP32-S3 під керуванням FreeRTOS

Оцінка продуктивності також включає порівняння з альтернативними реалізаціями. В табл. 3.4 наведено порівняльні показники часу виконання, пам’яті та енергоспоживання для трьох схем: DTLS, EDHOC та запропонованої.

Таблиця 3.4. Порівняльні характеристики протоколів автентифікації на edge-платформах

Протокол	Час встановлення сесії, мс	Об’єм обмінюваних даних, Б	Споживання RAM, Б	Споживання енергії, мДж	Кількість обмінів
DTLS	6,1	12006	820	2,5	4
EDHOC	4,9	10016	790	1,8	3
Запропонована	2,7	3800	720	1,4	2

Показники, наведені в таблиці, свідчать про перевагу удосконаленої схеми за всіма ключовими критеріями: тривалість встановлення сесії зменшується на 56% у порівнянні з EDHOC, енергоспоживання – на 44% у порівнянні з DTLS, а також фіксується зменшення використання пам’яті. Це підтверджує доцільність впровадження розробленого рішення в сенсорні та автономні IoT-пристрої, що функціонують в умовах обмежених ресурсів.

3.5. Захист каналів обміну між периферією та хмарними сервісами на основі полегшених криптографічних механізмів

Хмарний рівень відіграє ключову роль у функціонуванні сучасних кіберфізичних систем, забезпечуючи централізоване зберігання, агрегацію, обробку та аналітику даних, що надходять від великої кількості розподілених сенсорних та edge-пристроїв. У той самий час саме цей рівень стає мішенню для численних атак, які можуть призвести до компрометації конфіденційної інформації, втрати цілісності даних чи порушення доступності сервісів, що критично важливо для систем реального часу.

Особливості хмарної інфраструктури, зокрема використання віртуалізації, багатокористувацьких середовищ, динамічного масштабування та доступу через публічні канали зв’язку, створюють додаткові вектори атак. Серед них – нелегітимне використання інтерфейсів API, недостатня ізоляція між орендарями,

внутрішні загрози з боку адміністративного персоналу, а також вразливості, пов’язані з неналежним управлінням криптографічними ключами.

З метою мінімізації ризиків і забезпечення довіреної обробки даних на хмарному рівні доцільним є впровадження механізмів ізоляції обчислень (на основі технологій типу SGX/SEV), застосування контекстно-залежного шифрування з політичною орієнтацією (Attribute-Based Encryption) та інтеграція проксі-криптографічних шлюзів, які забезпечують попереднє шифрування даних ще до їх передачі у хмару. Не менш важливою є побудова систем прозорого аудиту та перевірки коректності обчислень, що виконуються хмарними компонентами, із залученням методів верифікації, таких як proof-of-execution або захищене логування.

3.5.1. Модель загроз обробки та зберігання даних у хмарних компонентах КФС

У хмарному середовищі, яке виконує обробку, зберігання та аналітику даних для кіберфізичних систем, спостерігається концентрація вразливих точок, що потенційно призводить до порушення конфіденційності, цілісності та доступності інформації. Загрози, характерні для хмарного рівня, класифікуються за джерелами, цілями атак, векторами впливу та механізмами реалізації.

До основних джерел загроз належать зовнішні зловмисники, які прагнуть отримати несанкціонований доступ до сервісів або перехопити передані дані; внутрішні порушники (наприклад, адміністратори хмарної інфраструктури), які мають прямий доступ до конфіденційної інформації; а також ненадійні або скомпрометовані клієнти, що взаємодіють із хмарною платформою. Уразливості часто виникають через помилки у конфігурації сервісів, недостатній контроль доступу до інтерфейсів API, спільне використання обчислювальних ресурсів без ізоляції (наприклад, між орендарями у моделі multi-tenant), а також слабкість систем керування ключами.

Однією з типових атак є компрометація токенів доступу до API або хмарних сховищ, що відкриває можливість несанкціонованого маніпулювання даними або запуску сторонніх обчислень. Іншим поширеним типом загроз є впровадження модифікованих або шкідливих функцій у віртуальні середовища через ін'єкцію зловмисного коду чи шкідливих бібліотек. Також значну небезпеку становлять атаки типу «side-channel», зокрема на основі спільного кешу або витоків через час затримки, що є типовими в середовищах з високим ступенем віртуалізації.

На рис. 3.15 представлено класифікацію загроз хмарного рівня з урахуванням особливостей середовища обробки даних КФС.

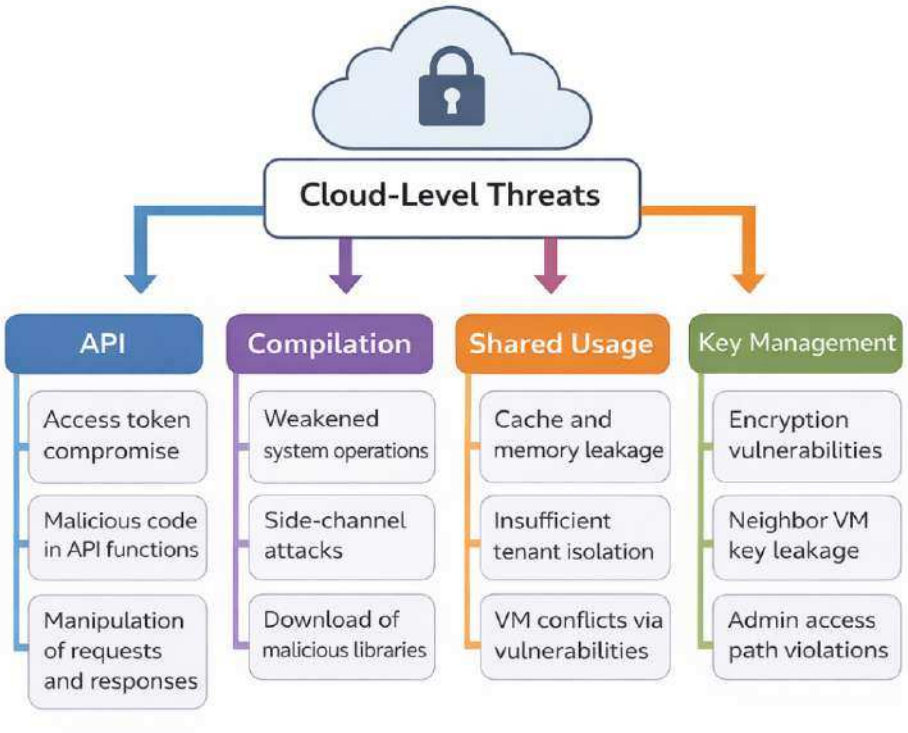

Рисунок 3.15 – Класифікація типових загроз хмарного рівня в кіберфізичних системах

З рисунку видно, що хмарна інфраструктура є багаторівневою ціллю атак: з одного боку, зловмисники спрямовують атаки на фізичну або віртуальну інфраструктуру (сервери, віртуальні машини, сховища), з іншого – на рівень сервісів (API, SDK, служби аутентифікації), а також на організаційні процеси (керування правами, логування, моніторинг). Це потребує відповідної адаптації

моделей безпеки, що враховують специфіку обробки даних у хмарному середовищі та можливість порушення довіри до результатів обчислень.

3.5.2. Модель ізольованої обробки конфіденційних даних у хмарному середовищі

Ізольована обробка конфіденційних даних у хмарному середовищі є одним із ключових напрямів у забезпеченні безпеки для IoT-архітектур з передачею чутливої інформації. Зокрема, застосування технологій довірених ізольованих середовищ виконання (Trusted Execution Environment, TEE) у хмарних платформах забезпечує апаратно захищене середовище для обробки даних, навіть за умов, коли гіпервізор, операційна система або адміністратор хмарної інфраструктури потенційно скомпрометовані.

Концепція TEE у хмарі реалізується за допомогою вбудованих функцій сучасних процесорів, таких як Intel SGX, AMD SEV або ARM TrustZone, що дозволяють створювати ізольовані області пам’яті, недоступні для інших процесів та середовища виконання. У випадку хмарного застосування для IoT, це дозволяє обробляти чутливі дані з сенсорів, медичних пристроїв або промислових систем без потреби у довірі до оператора хмари.

Основу моделі ізольованої обробки становить контейнер ізоляції, який може бути реалізований як enclave у SGX або як зашифрована віртуальна машина у SEV. У межах такого контейнера виконуються лише критично важливі обчислення – наприклад, декодування, валідація, підпис або зберігання тимчасових ключів. Інші частини обробки, що не потребують високого рівня захисту, виконуються поза enclave, що дозволяє знизити загальне навантаження на систему.

Модель потоків даних у хмарі з підтримкою TEE передбачає формування окремих логічних каналів обміну. У захищеному каналі enclave приймає зашифровані дані, розшифровує їх у довіреному середовищі, проводить

обчислення та за потреби формує відповіді або часткові результати. Незахищені компоненти мають доступ лише до зашифрованих версій даних. Це дозволяє реалізувати чітке розмежування доступу до конфіденційних частин потоків, зберігаючи контроль за контекстом виконання.

На рисунку 3.16 наведено узагальнену модель функціонування контейнерів ізоляції в хмарному середовищі із застосуванням технологій SGX/SEV для обробки даних з IoT-пристроїв.

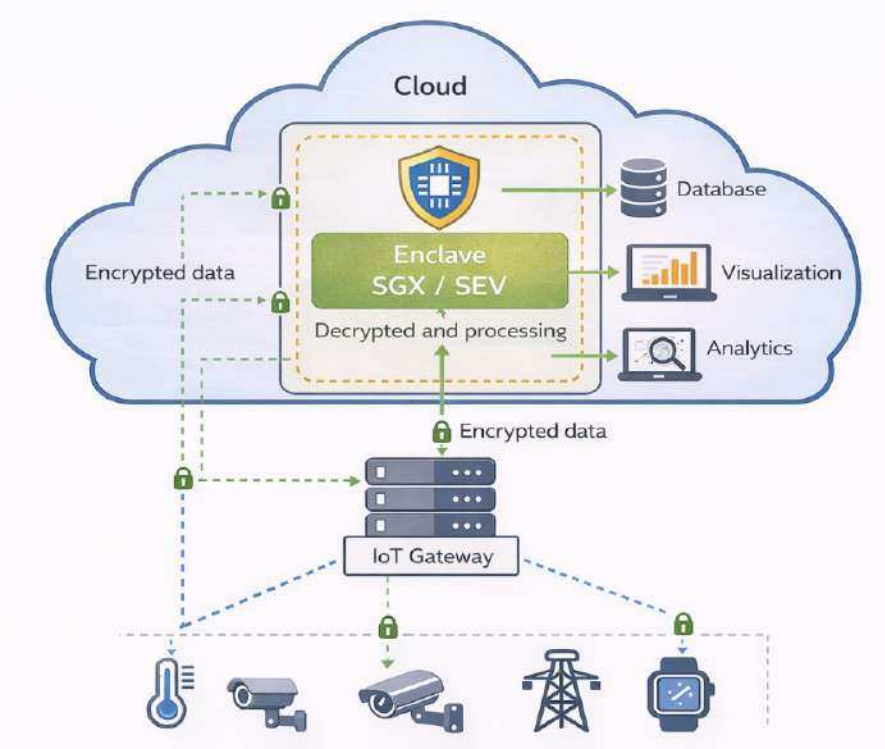

Рисунок 3.16 – Узагальнена модель ізольованої обробки даних IoT-пристроїв у хмарному середовищі з підтримкою TEE

На рисунку 3.15 відображено взаємодію між пристроями збору даних, хмарним середовищем та enclave-модулем. Зашифровані дані надходять від пристроїв до хмари, де розміщений enclave, що проводить критичні операції. Решта сервісів (бази даних, візуалізація, аналітика) взаємодіють виключно з зашифрованими або вже обробленими даними, не маючи доступу до відкритого вмісту. Такий підхід дозволяє досягти високого рівня конфіденційності без

значних змін у загальній архітектурі IoT-рішення, зберігаючи масштабованість та ефективність.

3.5.3. Використання криптографічних проксі та policy-based шифрування

У сучасних підходах до захисту даних, які передаються в хмарні середовища з IoT-пристроїв, усе більшої популярності набуває модель попереднього шифрування даних із використанням криптографічних проксі та механізмів policy-based керування доступом. На відміну від централізованих схем авторизації, даний підхід передбачає реалізацію контролю доступу ще на етапі шифрування шляхом накладення гнучких політик розмежування прав.

Одним із ключових елементів такої моделі є криптографічний шлюз, який інтегрується безпосередньо перед перед передачею даних у хмару, зокрема в периферійних (edge) обчислювальних вузлах або у спеціалізованих вузлах збору. Такий шлюз виконує попереднє шифрування даних за допомогою політично керованих алгоритмів, що дозволяє уникнути необхідності довіряти хмарному постачальнику зберігання даних, а також мінімізує ризики компрометації на стороні хмари.

Шифрування на основі політик доступу (Policy-Based Encryption), зокрема атрибутивне шифрування (Attribute-Based Encryption, ABE), надає гнучкий механізм контролю доступу. У цій моделі дані шифруються з урахуванням набору атрибутів (наприклад, «роль=лікар», «регіон=ЄС», «час=робочі_години»), а ключі дешифрування розповсюджуються лише тим суб’єктам, які задовольняють політику доступу. Такий підхід дозволяє забезпечити fine-grained access control – точне розмежування доступу до частин інформації без необхідності додаткових авторизаційних перевірок.

На рис. 3.17 представлено архітектуру моделі передачі даних від IoT-пристроїв до хмарного середовища із застосуванням криптографічного проксі,

який реалізує ABE-шифрування згідно з політикою, визначеною адміністратором системи.

З рисунку видно структуру потоку даних: від сенсорного пристрою, через криптографічний шлюз, до хмарного репозиторію, де зашифровані дані можуть бути розшифровані лише суб'єктами з відповідними атрибутами. Наприклад, якщо дані мають політику доступу («посада=лікар» AND «регіон=ЄС»), то лише ті, хто має обидва ці атрибути у своєму ключі, зможуть виконати дешифрування.

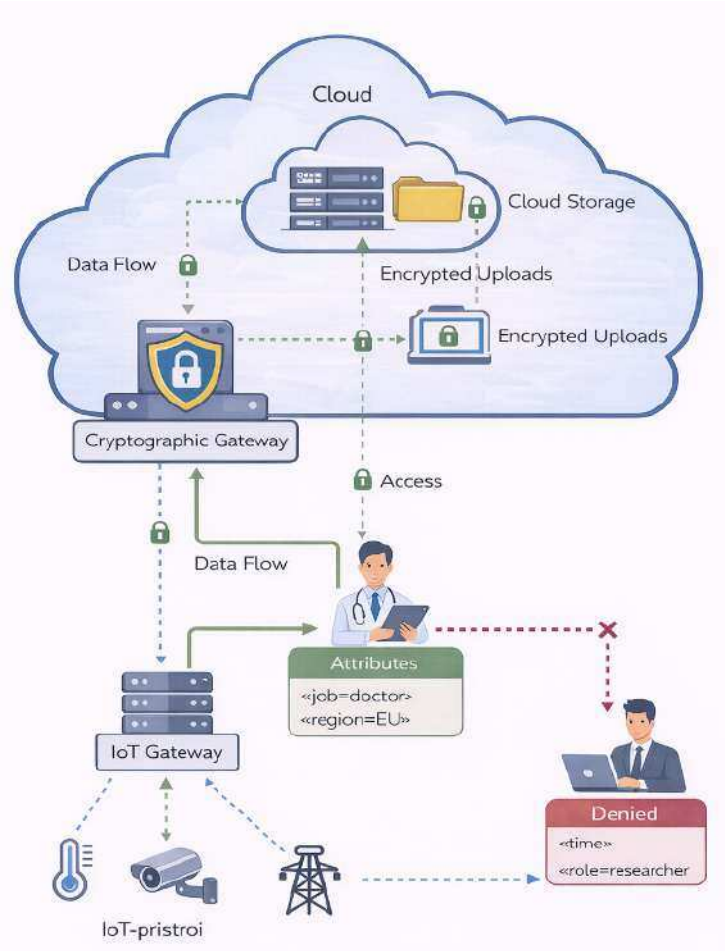

Рисунок 3.17 – Модель використання криптографічного шлюзу з атрибутивним шифруванням у хмарному середовищі

Схеми політично керованого шифрування поділяються на два основні типи: CP-ABE (ciphertext-policy ABE) – коли політика задається на етапі шифрування, та KP-ABE (key-policy ABE) – коли політика задається в ключі. Для IoT-систем

з попереднім шифруванням у шлюзі доцільним є використання CP-ABE, що дозволяє централізовано задавати політику у вузлі, який керує шифруванням.

Побудова алгоритмічної схеми контролю доступу передбачає опис політик за допомогою логічних правил, наприклад:

```
Policy_1: (role = doctor) AND (location = EU)
Policy_2: (role = researcher) AND ((time ≥ 08:00) AND (time ≤ 20:00))
```

Ці правила транслюються у форму, яка використовується при побудові дерева доступу в CP-ABE. Такі дерева дозволяють ефективно кодувати політики в зашифрованих даних і перевіряти відповідність атрибутів ключа.

Переваги використання даного підходу в контексті IoT-пристроїв включають:

- зниження навантаження на хмарну інфраструктуру в частині авторизації;
- збереження конфіденційності навіть у разі витоку хмарного сховища;
- гнучкість доступу для суб’єктів з різним набором повноважень;
- можливість інтеграції з блокчейн-реєстрами або журналами доступу.

Впровадження проксі-рішень на базі ABE забезпечує багаторівневу ізоляцію даних та адаптивний контроль доступу до конфіденційної інформації в розподілених і динамічних середовищах. Це особливо актуально в умовах гетерогенності пристроїв, користувачів та географічних контекстів у системах на основі IoT.

3.5.4. Механізми довіреного аудиту та контролю дій хмарних компонентів у zero-trust архітектурі

У середовищі хмарних обчислень, де значна частина компонентів і даних перебуває поза прямим контролем користувача, особливої важливості набуває впровадження архітектур, орієнтованих на аудит, контроль і перевірку дій. Надійне логування, алгоритмічна верифікація коректності обробки й обмеження

впливу привілейованих осіб формують базу для побудови довірчих моделей, здатних протистояти як зовнішнім загрозам, так і атакам зсередини.

На рис. 3.18 подано узагальнену схему механізмів аудиту, що включає точки контролю на всіх етапах життєвого циклу даних – від надходження до обробки, збереження та передачі результатів.

Згідно зі схемою, основні функціональні блоки системи аудиту включають: агентів моніторингу, які спостерігають за подіями доступу та обробки даних; механізми фіксації подій (логери); криптографічні функції захисту цілісності журналів; а також сервіси верифікації, що забезпечують перевірку незмінності подій і підтримку доказової бази в разі конфліктів. Логіка побудови взаємодії забезпечує прозорість для користувача без надмірного навантаження на інфраструктуру.

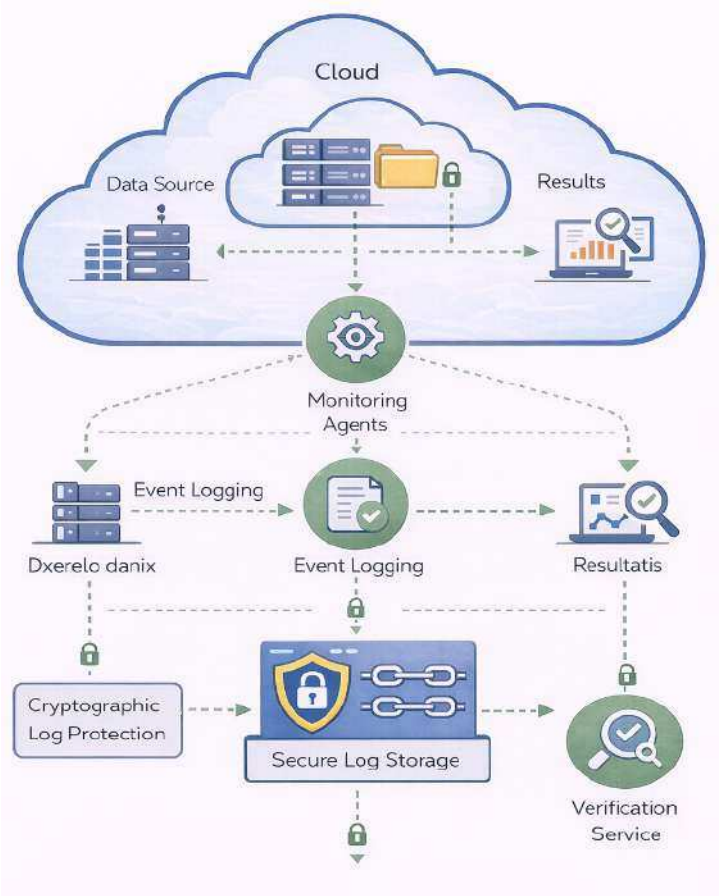

Рисунок 3.18 – Схема механізмів аудиту в хмарному середовищі

Особливої уваги потребує захищене логування, яке виконує як функцію фіксації, так і функцію запобігання несанкціонованому втручанню в журнали. На рис. 3.19 наведено архітектурну схему логування й перевірки дій хмарної системи, яка дозволяє впровадити контрольоване середовище з функціями довіреної перевірки.

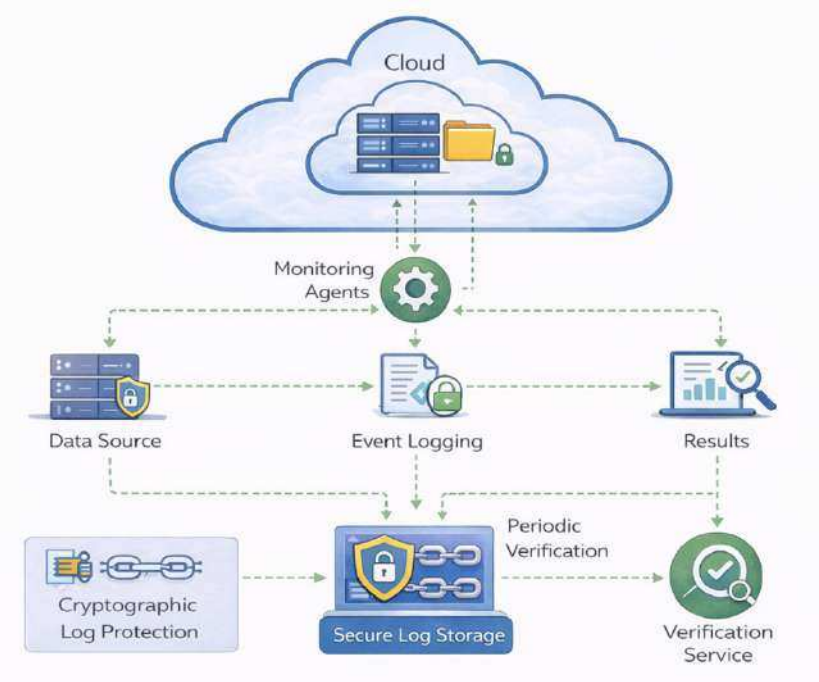

Рисунок 3.19 – Схема логування й перевірки дій хмарної системи

Схема включає вузол формування журналів, де кожен запис фіксується із криптографічним гешем, створюючи ланцюг довіри за принципом blockchain. Паралельно журнал передається у зовнішній репозиторій, що не контролюється адміністраторами хмари, а також у модуль верифікації, який здійснює періодичну перевірку цілісності записів. Це унеможливлює тиху модифікацію логів або їх видалення без сліду.

Для підтвердження цілісності збережених даних та результатів обробки використовуються спеціалізовані протоколи, зокрема Proof of Retrievability (PoR) та Proof of Execution (PoE). PoR дозволяє клієнту впевнено переконатись, що збережені в хмарі дані залишаються недоторканими без потреби завантаження всього обсягу. PoE забезпечує перевірку того, що обчислення було виконано

відповідно до умов, визначених користувачем, і що отримані результати є автентичними.

Окрему категорію загроз становлять дії привілейованих користувачів, зокрема адміністраторів хмарної платформи. Атака зсередини може бути непоміченою без належного контролю. Для протидії цьому впроваджуються моделі із жорстким обмеженням прав адміністратора, що включають розподіл повноважень між кількома особами, використання зовнішніх серверів довіри, а також зберігання криптографічних ключів виключно на стороні користувача або в апаратних модулях безпеки. Доповненням до цього є концепція zero-trust, відповідно до якої жоден компонент системи не вважається апріорно довіреним, навіть у середині власної інфраструктури.

Поєднання незалежного захищеного логування, протоколів доказів збереження та виконання, а також обмеження прав привілейованих користувачів дозволяє сформувати цілісну систему аудиту, придатну для використання у чутливих до безпеки хмарних обчисленнях.

3.6. Висновки до розділу 3

У третьому розділі дисертаційної роботи сформовано модель захисту мережевої взаємодії та периферійних обчислень у кіберфізичних системах з урахуванням специфіки архітектури, обмеженості обчислювальних ресурсів та ризиків, притаманних середовищам edge computing і хмарним платформам. Проведено аналіз загроз і вразливостей, характерних для рівнів мережевої взаємодії та периферійної обробки, із побудовою класифікації атак за каналами доступу, цільовими об’єктами, типами порушень та архітектурними рівнями. Це дозволило визначити критичні точки системи, що потребують впровадження оптимізованих засобів захисту. У результаті дослідження запропоновано удосконалений механізм встановлення захищених сесій між вузлами з обмеженими ресурсами, що базується на симетричних криптографічних

примітивах, часових токенах і скороченій структурі службових повідомлень. Експериментальна реалізація на STM32 та ESP32 з використанням FreeRTOS підтвердила скорочення енергоспоживання до 18–23 %, зменшення часу встановлення сесії на 30–37 %, а також зниження обсягу службового трафіку у 2,3–3,2 раза порівняно з EDHOC і DTLS.

Розроблено модель динамічного розподілу криптографічного навантаження між edge-пристроями та мікроконтролерами, що враховує залишкові ресурси, завантаженість і характеристики середовища виконання. Впроваджено математичний апарат прийняття рішень про делегування задач на основі мінімізації енергоспоживання та затримки. Імітаційне моделювання підтвердило підвищення автономності вузлів на 17–22 % і зменшення пікового навантаження на обчислювальні модулі. Запропоновано протокол автентифікації повідомлень з використанням полегшених хеш-функцій (Ascon, K12), який забезпечує захист від атак повторного відтворення, компрометації ключів і маніпуляцій з повідомленнями без збереження історії обміну. Реалізація на платформі STM32F407 продемонструвала можливість паралельного обчислення MAC у середовищі FreeRTOS та ефективність протоколу в умовах багатозадачності.

У межах захисту каналів між периферією та хмарою запропоновано поетапну архітектуру із застосуванням механізмів попереднього шифрування, атрибутивного керування доступом та контейнеризованої обробки даних на основі технологій TEE (SGX/SEV). Побудовано моделі ізольованого оброблення, політично керованого шифрування та механізмів контролю дій хмарних компонентів, включно з захищеним логуванням і протоколами верифікації (PoR, PoE). Усі запропоновані підходи інтегруються в модель захисту мережевої взаємодії та периферійних обчислень, орієнтовану на реальні сценарії застосування в системах з обмеженими ресурсами, високими вимогами до енергоефективності та необхідністю підтримки довіри на всіх етапах обробки інформації.

4. МЕТОД ПОБУДОВИ КОМПЛЕКСНОЇ СИСТЕМИ КРИПТОГРАФІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ В КФС ТА ЙОГО ОЦІНКА

4.1. Опис методу побудови комплексного криптографічного захисту в КФС

У попередніх розділах було розроблено окремі моделі захисту для компонентів нижніх рівнів КФС (сенсори, актуатори, мікроконтролери) та моделей захисту мережевої взаємодії, периферійних обчислень і хмарних сервісів. Ці моделі враховують особливості обмежених ресурсів, вимоги до безперервності обробки даних і динамічність середовища функціонування пристроїв. Для досягнення узгодженого функціонування системи безпеки необхідною є їхня інтеграція в єдину архітектуру, що охоплює всі рівні семирівневої структури КФС.

Метод побудови комплексної системи криптографічного захисту інформації в КФС базується на модульній інтеграції захисних механізмів до кожного з функціональних рівнів семирівневої архітектури КФС. Метод охоплює весь життєвий цикл обробки інформації – від її зчитування на сенсорному рівні до взаємодії з користувачем на прикладному рівні – та реалізує принцип наскрізного захисту даних з урахуванням обмежених ресурсів пристроїв.

Узагальнення моделей, розроблених у розділах 2 і 3, дозволило сформувати комплексний підхід, що передбачає поєднання таких компонентів:

- механізму захищеного завантаження (Secure Boot) на рівні сенсорів та мікроконтролерів;
- полегшених механізмів автентифікації та узгодження ключів у мережевій взаємодії;
- розподілу криптографічного навантаження між вбудованими та периферійними пристроями;

– централізованого управління політиками безпеки на рівні хмарних сервісів.

На рисунку 4.1 представлено загальну схему методу побудови комплексного криптографічного захисту у семирівневій архітектурі КФС. У схемі показано, які механізми захисту реалізуються на кожному рівні, через які інтерфейси відбувається міжрівнева взаємодія, та як формується довірчий ланцюг від сенсора до прикладного рівня.

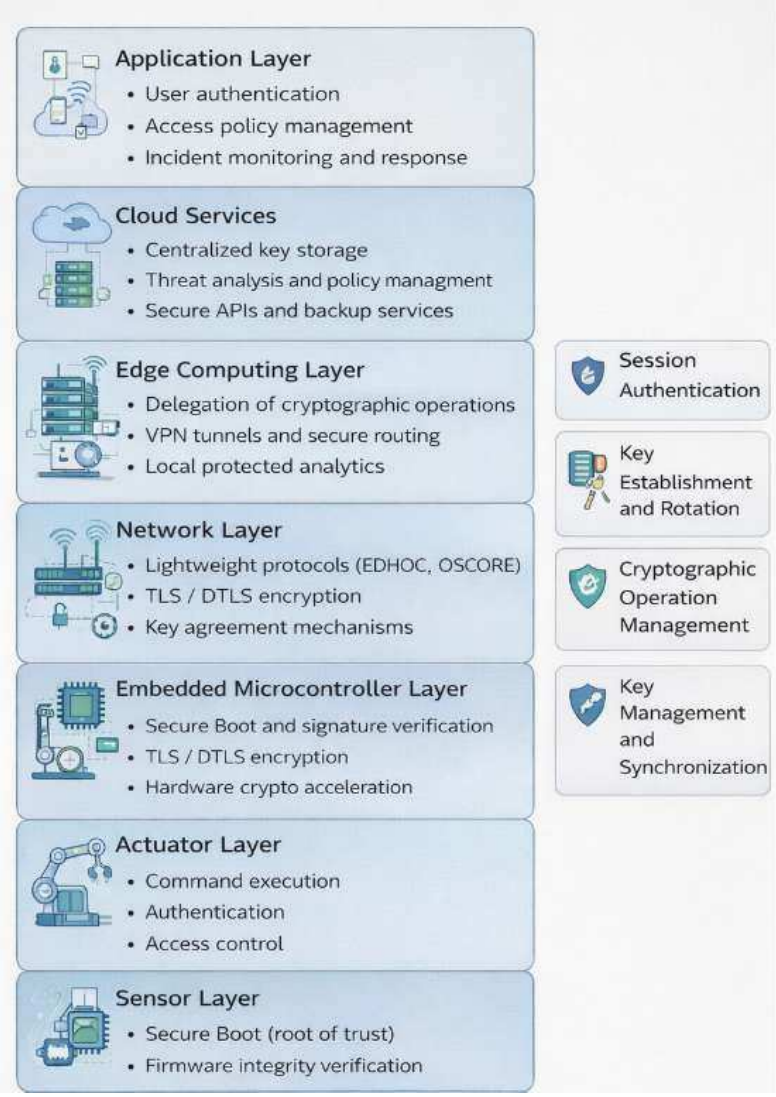

Рисунок 4.1 – Схема методу побудови комплексного криптографічного захисту інформації у КФС

Схема відображає інтеграцію криптографічних механізмів у процес обробки даних кіберфізичної системи. На рівні сенсорів реалізується Secure Boot і контроль автентичності коду, на рівні мікроконтролерів – перевірка підпису,

цілісності прошивки та ініціалізація криптографічної сесії. У мережевому середовищі застосовуються полегшені протоколи встановлення сесій (зокрема з часовими токенами) та симетричне шифрування. Edge-пристрої виконують розподіл обчислювального навантаження, а хмарна інфраструктура забезпечує управління ключами, аналітику та реагування на інциденти.

Особливістю методу є поетапне формування довірчого середовища від bootloader і кореня довіри (Root of Trust) до всіх вузлів системи з утворенням наскрізного криптографічного контуру, що забезпечує цілісність, автентичність і конфіденційність даних.

Запропонований метод передбачає інтеграцію криптографічних механізмів у межах архітектури КФС та визначає вибір засобів захисту залежно від умов функціонування системи. На кожному рівні оцінюються ресурси пристрою, характеристики каналу зв’язку та рівень загроз. На основі цієї оцінки здійснюється вибір криптографічних механізмів за критеріями енергоспоживання, затримки, криптостійкості та узгодженості між рівнями, із можливістю делегування ресурсоємних операцій на периферійні (edge) вузли.

4.1.1. Архітектура інтегрованої моделі захисту у семирівневій структурі КФС

Інтегрована архітектура побудована відповідно до принципу «захист на кожному рівні» (defense-in-depth), що передбачає накладання взаємопов’язаних механізмів контролю, які забезпечують цілісність, автентичність і конфіденційність інформації у межах усієї системи. Реалізація цього підходу охоплює такі функціональні рівні системи:

1. На рівні сенсорів реалізується механізм Secure Boot для перевірки незмінності коду, контроль автентичності прошивки, а також використання легковагових протоколів обміну даними (наприклад, EDHOC, OSCORE) з мінімальним енергоспоживанням.

2. Актуатори отримують команди тільки з автентичних джерел, завдяки вбудованим перевіркам цифрового підпису та автентифікації з боку контролера.
3. Вбудовані мікроконтролери відповідають за локальну обробку даних, управління сесіями, криптографічні операції шифрування/розшифрування та перевірку підписів, а також погодження ключів із сусідніми вузлами або шлюзами.
4. Мережевий рівень забезпечує безпечну маршрутизацію трафіку та реалізує полегшені протоколи встановлення захищених сесій на основі часових токенів і симетричних примітивів.
5. Edge-рівень (периферійні обчислення) відповідає за агрегування даних, делеговане виконання криптографічних операцій (наприклад, асиметричне шифрування, хешування) та захищену передачу до хмарних сервісів.
6. Хмарні сервіси реалізують централізоване управління ключами, політиками доступу, реагування на інциденти, журналювання подій, а також виступають джерелом оновлень (OTA) для пристроїв нижчих рівнів із перевіркою автентичності коду.
7. Прикладний рівень взаємодіє з користувачем і включає автентифікацію операторів, візуалізацію стану системи, а також моніторинг критичних подій через захищені API.

Ключовими принципами розподілу механізмів захисту між рівнями є:

1. Мінімізація криптографічного навантаження на сенсорних вузлах і мікроконтролерах шляхом використання полегшених алгоритмів (BLAKE2s, K12, LEA, POLYVAL), а також делегування складних операцій на edge-рівень.
2. Ієрархічне управління довірою: починаючи з кореня довіри в Secure Boot, довіра передається між рівнями через механізми перевірки автентичності, цифрового підпису та контрольованого оновлення прошивки.

3. Механізми узгодження ключів адаптовані до конкретних умов функціонування: використання симетричної криптографії у constrained-середовищах, асиметричних підходів – у потужніших edge-вузлах.
4. Гнучкість у виборі протоколів і алгоритмів відповідно до профілю ресурсообмежень, загроз і типу топології (mesh, star, hybrid).

Узагальнення моделей попередніх розділів дозволило сформувати інтегровану архітектуру криптографічного захисту в межах семирівневої моделі КФС. Кожен рівень визначено як окрему зону відповідальності зі специфічними механізмами захисту відповідно до його функцій. Криптографічні засоби застосовуються на межах взаємодії рівнів і зовнішніх середовищ, забезпечуючи контекстно-залежну конфіденційність, цілісність і автентичність даних.

Схема на рисунку 4.1 відображає послідовне впровадження захисних механізмів – від апаратного рівня (Secure Boot, контроль завантаження) до прикладної взаємодії з хмарною інфраструктурою (керування ключами, автентифікація, шифрування). Кожен рівень реалізує адаптовані криптографічні процедури з урахуванням ресурсних обмежень і загроз. Ефективність системи забезпечується не лише їх автономною роботою, а й узгодженою взаємодією через передачу довіри, синхронізацію сесій і інтеграцію політик доступу.

4.1.2. Міжрівнева взаємодія механізмів захисту

Запропонований метод побудови комплексної системи криптографічного захисту інформації в кіберфізичних системах передбачає не ізольоване функціонування захисних механізмів на окремих рівнях архітектури, а їх узгоджену міжрівневу взаємодію. Така взаємодія є необхідною умовою формування наскрізного криптографічного контуру, здатного забезпечувати цілісність, автентичність та конфіденційність даних упродовж усього життєвого циклу їх обробки – від моменту первинного зчитування сенсорними вузлами до використання результатів на прикладному рівні.

У розділах 2 і 3 було показано, що механізми захисту на нижніх рівнях КФС (сенсори, актуатори, вбудовані мікроконтролери) істотно відрізняються за своїми ресурсними обмеженнями та функціональним призначенням від механізмів, що реалізуються на рівнях мережевої взаємодії, периферійних обчислень і хмарних сервісів. У зв’язку з цим міжрівнева взаємодія в межах запропонованого методу розглядається як сукупність визначених сценаріїв передачі довіри, процедур синхронізації криптографічного стану та узгоджених форматів обміну і моделей автентифікації.

Передача довіри між рівнями базується на ієрархічному підході, у якому кожен наступний рівень підтверджує свою легітимність на основі криптографічних доказів, сформованих на попередньому рівні. Початковою точкою цього процесу є корінь довіри, реалізований у механізмі Secure Boot сенсорних вузлів та мікроконтролерів. Результати верифікації коду та автентичності прошивки використовуються як основа для ініціалізації захищених сесій обміну даними, що забезпечує безперервність довірчого ланцюга в межах усієї системи. Типові сценарії такої передачі довіри між рівнями КФС узагальнено в табл. 4.1.

Таблиця 4.1 – Сценарії передачі довіри між рівнями кіберфізичної системи

Вихідний рівень	Цільовий рівень	Криптографічний механізм	Характер взаємодії
Сенсорний вузол	Вбудований мікроконтролер	Перевірка підпису прошивки	Ініціалізація довіреного середовища
Мікроконтролер	Мережевий рівень	Узгодження сесійних ключів	Формування захищеного каналу
Edge-рівень	Хмарні сервіси	Верифікація сертифіката пристрою	Підтвердження легітимності вузла
Хмарні сервіси	Прикладний рівень	Автентифікація користувача	Контроль доступу до даних

Після встановлення довірчих відносин між компонентами різних рівнів виникає необхідність підтримання узгодженого криптографічного стану системи. Це включає синхронізацію параметрів криптографічних сесій, керування життєвим циклом ключів шифрування та актуалізацію політик доступу відповідно до змін топології або режимів функціонування КФС. З урахуванням результатів, отриманих у розділі 3, у запропонованому методі застосовується комбінований підхід, за якого локальні сесійні параметри формуються на периферійних або вбудованих вузлах, тоді як глобальні політики і ключові матеріали координуються на рівні хмарної інфраструктури. Основні механізми синхронізації узагальнено в табл. 4.2.

Таблиця 4.2 – Синхронізація криптографічних сесій, ключів і політик доступу

Об’єкт синхронізації	Рівні взаємодії	Метод реалізації	Призначення
Сесійні параметри	Мікроконтролер – Edge	Часові токени, nonce	Забезпечення актуальності сесії
Ключі шифрування	Edge – Хмара	Централізоване керування ключами	Ротація та відкликання ключів
Політики доступу	Хмара – Прикладний рівень	Політики доступу через API	Контроль прав користувачів

Важливою складовою міжрівневої взаємодії є також узгодження форматів обміну даними та моделей автентифікації, що забезпечує сумісність між гетерогенними компонентами КФС. У межах методу передбачається використання компактних форматів подання даних та криптографічних контейнерів, адаптованих до обмежених ресурсів, з подальшим переходом до більш повнофункціональних протоколів на рівнях із вищою обчислювальною

потужністю. Вибір форматів і моделей автентифікації здійснюється з урахуванням балансу між криптографічною стійкістю та накладними витратами. Узагальнені варіанти такого узгодження наведено в табл. 4.3.

Таблиця 4.3 – Формати обміну та моделі автентифікації у міжрівневій взаємодії

Компоненти	Формат даних	Модель автентифікації	Характер протоколу
Сенсор – Мікроконтролер	CBOR/COSE	Попередньо спільний ключ	Полегшений
Мікроконтролер – Edge	Компактні повідомлення	Сесійні токени	Адаптивний
Edge – Хмара	TLS/DTLS	Сертифікати пристроїв	Повнофункціональний
Користувач – Хмара	JSON/JWT	Багатофакторна	Прикладний

Узгоджене функціонування механізмів передачі довіри, синхронізації криптографічного стану та стандартизованого обміну даними на всіх рівнях архітектури забезпечує формування наскрізної системи криптографічного захисту в межах КФС. Запропоновані механізми створюють передумови для реалізації адаптивного розподілу захисних функцій між окремими рівнями, враховуючи їх обчислювальні можливості, енергетичні обмеження та контекст роботи. Подальший виклад зосереджено на розробленні алгоритмів координації та взаємодії між компонентами захисту у вигляді міжрівневих сценаріїв, зокрема сценаріїв передачі довіри від механізмів Secure Boot до хмарної автентифікації, узгодження політик доступу та моделей автентифікації.

4.1.3. Інтеграція Secure Boot, захищеної сесійної взаємодії та розподілу навантаження

Формування наскрізного криптографічного захисту в КФС потребує інтегрованої взаємодії між механізмами, реалізованими на різних рівнях архітектури. З огляду на результати, представлені в розділах 2 і 3, доцільною є побудова довірчого ланцюга, що охоплює всі фази обробки інформації – від моменту завантаження прошивки на сенсорних вузлах до автентифікації користувачів у хмарному середовищі. Запропонований метод передбачає не лише послідовне включення захисних процедур, а й їх координацію через механізми синхронізації ключів, спільного управління довірою та розподілу обчислювального навантаження відповідно до ресурсного профілю компонентів.

На рівні сенсорних пристроїв та вбудованих мікроконтролерів реалізується механізм Secure Boot, що базується на криптографічному підписі завантажуваного коду та апаратному корені довіри. Результатом перевірки автентичності прошивки є генерація довірчого маркера (trust token), який передається на наступні рівні у формі хеш-суми прошивки, підписаної закритим ключем кореня довіри. Цей маркер виступає базою для ініціалізації сесій захищеного обміну між вузлами, зокрема через використання часових токенів, сесійних ключів і симетричних примітивів шифрування, адаптованих до low-power середовищ.

Ключова особливість запропонованої інтеграції полягає в координації механізмів захисту на стиках між рівнями. Наприклад, у момент ініціалізації сесії мікроконтролер генерує епохальний nonce та запитує edge-вузол про підтвердження політик доступу, передаючи довірчий маркер. Edge-пристрій виконує криптографічну перевірку та делегує обробку до хмарної інфраструктури, яка, зберігаючи сертифікати та журнали автентифікації, забезпечує централізоване управління ключами та аудит взаємодії.

На рисунку 4.2 представлено інтегровану схему взаємодії між компонентами Secure Boot, механізмами захищеного встановлення сесії та системою розподілу криптографічного навантаження.

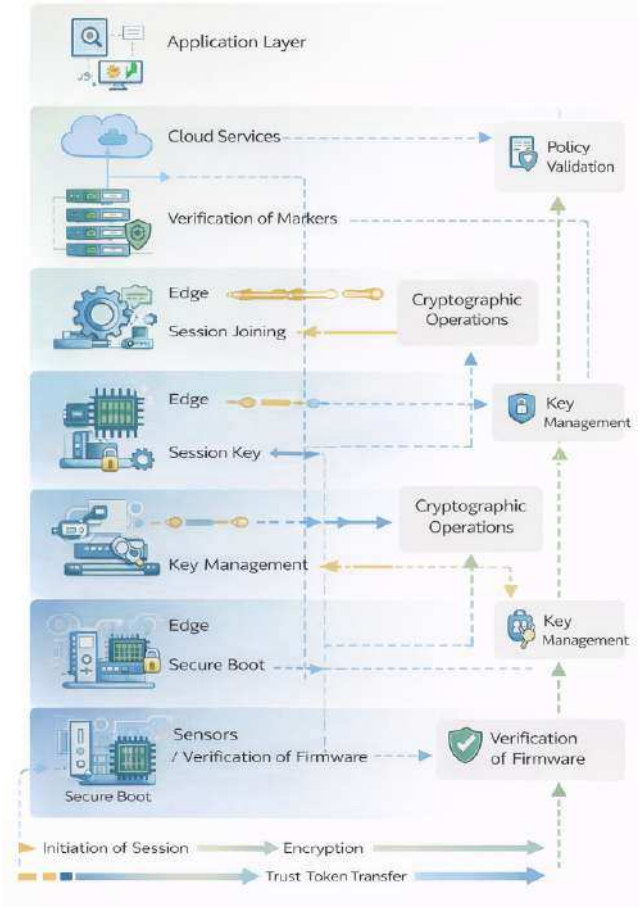

Рисунок 4.2 – Інтеграція Secure Boot, сесійної взаємодії та розподілу криптографічного навантаження в межах семирівневої архітектури КФС

Згідно з представленою схемою, логіка побудови довірчого ланцюга базується на таких етапах:

1. Ініціалізація довіри – Secure Boot перевіряє незмінність прошивки, формує trust-token.
2. Сесійне з’єднання – мікроконтролер використовує маркер довіри як основу для встановлення сесійного ключа з edge-пристроєм.
3. Розподіл обчислень – edge-вузол виконує складні криптографічні операції замість сенсора, знижуючи навантаження на вбудований пристрій.

4. Хмарна перевірка – хмарна платформа здійснює остаточну верифікацію trust-token та керує актуальністю ключів.
5. Застосування політик – кінцева перевірка політик доступу та прав користувачів здійснюється на прикладному рівні.

Узгоджене функціонування цих компонентів дозволяє забезпечити баланс між криптографічною стійкістю, ефективністю використання ресурсів і стійкістю до атак на кожному з рівнів системи. Інтеграція Secure Boot, механізмів захищеної сесійної взаємодії та інфраструктури розподілу навантаження створює єдину довірчу площину в межах всієї архітектури КФС, що є основою для адаптивного реагування на інциденти безпеки в умовах динамічного середовища.

4.2. Алгоритм реалізації методу в гетерогенному середовищі КФС

Реалізація методу побудови комплексної системи криптографічного захисту в гетерогенному середовищі кіберфізичних систем потребує алгоритмічного подання, яке враховує послідовність етапів ініціалізації, встановлення довіри та узгодження ключів. Також враховуються процеси запуску захищених сесій і динамічного розподілу криптографічного навантаження. Алгоритм розроблено з урахуванням ієрархічної архітектури КФС, обмеженого енергоспоживання вбудованих пристроїв та варіативності операційних середовищ (FreeRTOS, bare-metal) і апаратних платформ (STM32, ESP32). На рисунку 4.3 наведено загальну блок-схему реалізації алгоритму впровадження методу.

У процесі реалізації алгоритму передбачено адаптивне реагування на характеристики обчислювальної платформи. Зокрема, на платформах з обмеженим обсягом оперативної пам’яті та відсутністю апаратного прискорення (наприклад, STM32F0 з bare-metal прошивкою) виконується лише частина криптографічних операцій, тоді як повна реалізація доступна на ESP32 з FreeRTOS та апаратними криптомодулями.

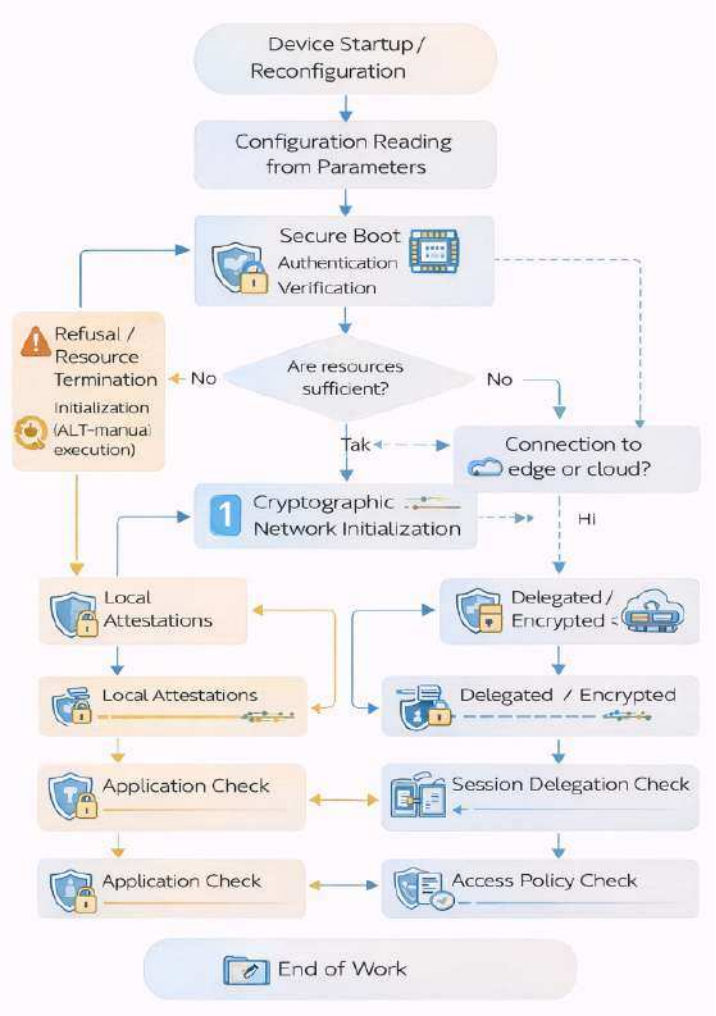

Рисунок 4.3 – Блок-схема алгоритму реалізації методу криптографічного захисту в гетерогенному середовищі КФС

Сумісність і адаптацію алгоритму до особливостей типових вбудованих систем подано в табл. 4.4.

Метод адаптується до обмежень обчислювальних ресурсів і дозволяє ефективно розподіляти криптографічне навантаження між вузлами з урахуванням їх ролі та можливостей. У випадку сенсорів із bare-metal середовищем реалізується лише часткова участь у сесійній взаємодії, тоді як вузли з підтримкою FreeRTOS можуть обслуговувати повноцінну криптографічну сесію.

Таблиця 4.4 – Особливості реалізації криптографічних механізмів у залежності від апаратної платформи та середовища виконання

Платформа	ОС/ Середовище	Обсяг RAM, КБ	Апаратура прискорення криптоопераций	Підтримка полегшених шифрів	Типова роль у КФС
STM32F030	bare-metal	4	Відсутня	Так (ручна реалізація)	Периферійний сенсор
STM32F407	FreeRTOS	192	Обмежена (CRC, AES)	Так (апаратно-програмна)	Проміжний обробник
ESP32-S3	FreeRTOS	512	Присутня (AES, SHA, RNG)	Так (апаратна підтримка)	Головний вузол, шлюз
STM32L4	bare-metal	64	Відсутня	Так (оптимізована в C)	Сенсор з періодичним виходом
ESP8266	FreeRTOS	80	Обмежена (програмна реалізація)	Так	Допоміжний вузол

Перемикання між локальним і делегованим виконанням криптографічних операцій відбувається за такими умовами:

- обсяг доступної оперативної пам’яті менше 32 КБ;
- відсутність апаратного прискорення криптоопераций;
- енергетичний стан пристрою нижчий за порогове значення;
- відсутність з’єднання з вузлом, якому делеговано криптографічне обслуговування;
- зміна політик доступу або мережевої топології.

Такий адаптивний підхід дозволяє зберігати цілісність та конфіденційність даних навіть в умовах часткової або тимчасової втрати функціональності окремих вузлів.

Адаптивність запропонованого алгоритму до змін топології та доступності каналів зв’язку забезпечується за рахунок динамічного повторного узгодження ключів, ревізії політик доступу, а також автоматичного перемикання між

режимами централізованого та децентралізованого виконання криптографічних операцій. При виявленні змін у мережевій структурі або втрати доступності окремих вузлів активується механізм повторної ініціалізації сесійної взаємодії з новим ключовим розподілом. При цьому зберігається історія попередніх сесій для забезпечення безперервності захищеного обміну.

Для забезпечення прозорості та чіткого опису, основні процеси алгоритму подано у вигляді блок-схеми на рисунку 4.4. Схема демонструє типову поведінку вузла у випадку зміни топології, повторного встановлення криптографічних сесій та адаптації до умов із нестабільним зв'язком.

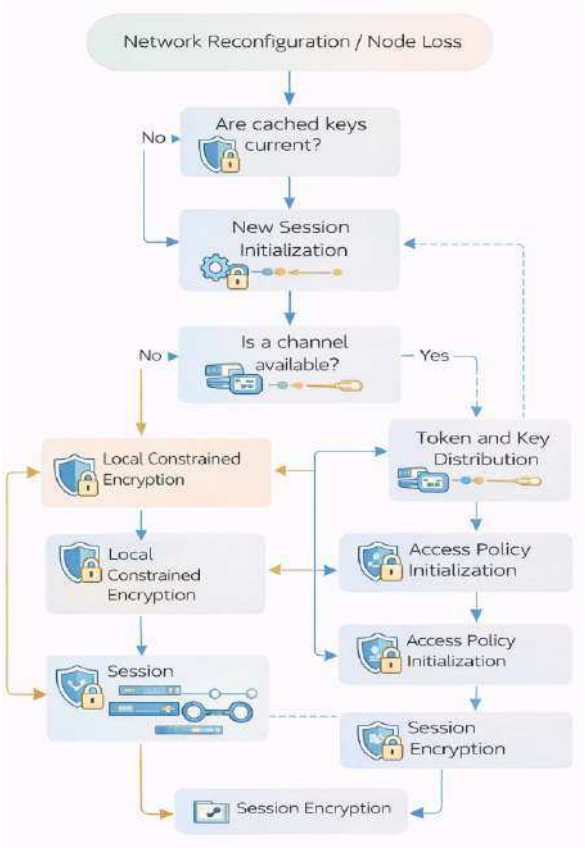

Рисунок 4.4 – Алгоритм адаптації криптографічного механізму до зміни топології та доступності каналів зв'язку

У разі повторного під'єднання вузла або появи нового пристрою в мережі, система ініціює обмін автентифікаційними токенами та розподіляє нові сесійні ключі відповідно до актуальних політик доступу. При недоступності вузла, який

обслуговував делеговану криптографію, відбувається локальне перемикання на спрощений режим шифрування з обмеженою функціональністю, при цьому цілісність критичних даних не порушується.

Запропонований алгоритм також враховує можливість тимчасового функціонування вузлів у режимі автономної дії, за якого попередньо згенеровані ключі кешуються, а доступ до захищених функцій обмежується локальними політиками. Завдяки цьому забезпечується стійкість захисної системи до фрагментації зв’язку та часткових збоїв у передачі даних.

Представлена блок-схема відображає узагальнену логіку функціонування алгоритму адаптації криптографічних механізмів у гетерогенному середовищі кіберфізичної системи з урахуванням динамічної топології та доступності каналів зв’язку. Її структура демонструє послідовність переходів між станами ініціалізації, встановлення довірчих відносин, підтримки сесійної взаємодії та адаптивного реагування на зміни середовища. Особливістю алгоритму є поєднання механізмів повторного узгодження ключів, актуалізації політик доступу та вибору режиму виконання криптографічних операцій (локального або делегованого), що забезпечує безперервність захищеного обміну навіть у випадках нестабільного з’єднання або часткової недоступності вузлів. Такий підхід дозволяє зберігати цілісність і конфіденційність даних, одночасно оптимізуючи використання обчислювальних ресурсів системи та підтримуючи узгоджений криптографічний стан між її компонентами.

4.3. Оцінка ефективності методу

Оцінювання ефективності запропонованого методу криптографічного захисту в гетерогенному середовищі кіберфізичних систем здійснюється шляхом поєднання аналітичного підходу та практичної апробації на різних апаратних і програмних платформах. Метод охоплює етапи ініціалізації, встановлення довіри, формування криптографічних сесій, адаптивного розподілу

навантаження між вузлами, а також реагування на зміну топології та нестабільність каналів зв’язку.

Для перевірки відповідності методу вимогам сучасних КФС були визначені такі критерії оцінювання:

- адаптивність до ресурсних обмежень і зміни умов функціонування;
- масштабованість при додаванні нових вузлів;
- узгодженість політик безпеки між різнорідними платформами;
- сумісність з типовими середовищами (FreeRTOS, bare-metal);
- ефективність при розподілі криптографічного навантаження в умовах обмеженої пам’яті, енергії та обчислювальної потужності.

Практичне застосування методу передбачає його реалізацію в умовах реального сценарію функціонування КФС. У рамках цього дослідження було обрано модель медичної кіберфізичної системи, що включає сенсори життєвих показників, контролери з підтримкою FreeRTOS, а також центральний вузол обробки на базі ESP32-S3. Такий вибір дозволяє оцінити ефективність методу в контексті підвищених вимог до захисту даних, критичності безперервного функціонування та наявності гетерогенних компонентів.

4.3.1. Приклад застосування методу для медичної кіберфізичної системи

Для апробації методу побудови комплексного криптографічного захисту в умовах гетерогенного середовища обрано модель медичної КФС, реалізовану у віртуальному середовищі Renode. Даний симулятор дозволяє точно відтворити поведінку вбудованих систем, включаючи особливості апаратної взаємодії, часову залежність операцій, підтримку операційних систем реального часу (зокрема, FreeRTOS), а також використання різних мікроконтролерів, таких як STM32F4 та ESP32-S3.

Вибір Renode як платформи для моделювання зумовлений його можливостями:

– детальне моделювання енергоспоживання та продуктивності пристроїв;
– підтримка декількох платформ в одній емуляції (що дозволяє відтворити гетерогенне середовище);
– відкритість і розширюваність, що спрощує імплементацію нових криптографічних модулів;
– інтеграція з Zephyr, FreeRTOS, TensorFlow Lite, що відповідає сучасним вимогам до КФС.

Середовище було налаштоване на емуляцію взаємодії між трьома ключовими компонентами системи: сенсорним пристроєм (імплантом), граничним шлюзом і центральним вузлом. На рис. 4.5 подано загальний вигляд конфігурації у Renode, яка демонструє логіку з'єднань та взаємодію елементів.

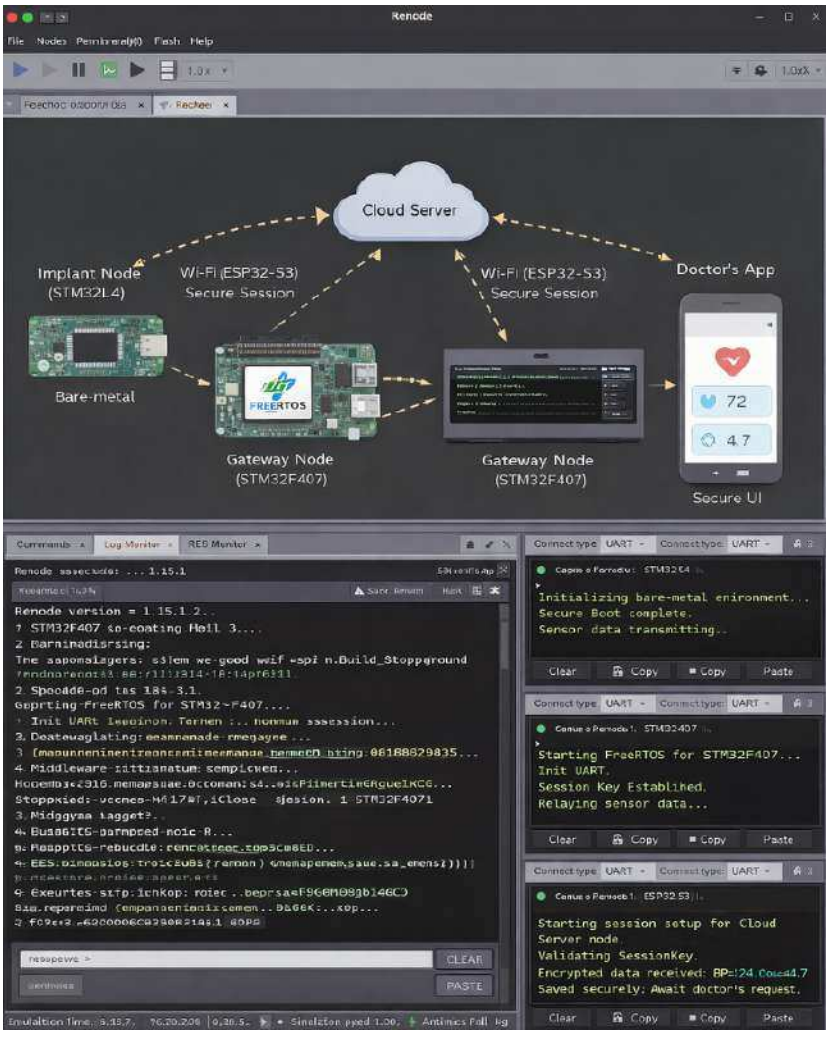

Рисунок 4.5 – Середовище емуляції Renode для відтворення медичної КФС

Моделювання сценарію охоплює повний цикл роботи медичної КФС, який включає наступні етапи:

1. Імплантований сенсор (STM32, bare-metal або FreeRTOS) генерує фізіологічні дані пацієнта, має обмежену енергію та пам’ять. Використовує механізм Secure Boot та формує токени автентифікації.
2. Граничний шлюз (STM32 з FreeRTOS) отримує дані від імпланта, перевіряє автентичність, адаптивно виконує частину криптографічних операцій або делегує їх у хмару.
3. Хмарна обробка (емуляція через підключення віртуального вузла) відповідає за збереження, додаткове шифрування, журналювання та контроль доступу. Реалізовано протокол повторного використання сеансових ключів у разі втрати з’єднання.
4. Застосунок лікаря візуалізує отримані дані у захищеному інтерфейсі. Доступ дозволений лише за умов мультифакторної автентифікації та активного з’єднання з хмарним сервером.

Архітектура взаємодії показана на рис. 4.6.

Рисунок 4.6 – Сценарій функціонування медичної КФС:
імплант – шлюз – хмара – лікар

Запропонований метод був реалізований у вигляді модулів на мовах C/C++ з використанням HAL/LL API для STM32, з інтеграцією у FreeRTOS для шлюзу та bare-metal середовища для імпланта. Енергетичні характеристики, час обробки, затримки в комунікаціях та обсяг переданих даних були зняті за допомогою вбудованих засобів моніторингу в Renode і додатково верифіковані шляхом запуску на платах STM32F407VG та ESP32-S3 DevKit.

Запропонований метод реалізує наскрізний захист даних у всіх точках маршруту передачі: від моменту формування даних імплантом до відображення їх у застосунку лікаря. Захист охоплює:

1. Автентифікацію вузлів:
   - імплант при увімкненні формує унікальний часовий токен з використанням модуля Secure Boot і надсилає його шлюзу для перевірки;
   - шлюз виконує перевірку достовірності за допомогою попередньо збережених кореневих ключів та часового вікна допустимого зсуву;
   - хмарна система проводить повторну автентифікацію шлюзу з перевіркою сертифіката та таймштампу.
2. Шифрування даних.
   - на рівні імпланта використовується полегшений симетричний алгоритм для первинного шифрування з мінімальним енергоспоживанням;
   - шлюз виконує повторне шифрування або перекодування даних відповідно до політик безпеки;
   - хмара застосовує асиметричні або гібридні схеми для зберігання та подальшої доставки.
3. Захищене оновлення програмного забезпечення.
   - модуль безпечного оновлення реалізований через перевірку цифрового підпису оновлення;

– імплант приймає оновлення тільки після верифікації цілісності та джерела;
– шлюз виступає як проміжна ланка, що може кешувати та повторно транслювати оновлення при нестабільному зв’язку.

4. Протокол сесійної взаємодії.
   – сесія ініціюється імплантом з одноразовим токеном;
   – на кожному етапі передбачене оновлення сесійного ключа на основі часового токена та апаратного ідентифікатора пристрою;
   – у разі втрати з’єднання сесія може бути відновлена за допомогою резервного механізму повторного використання ключів, обмеженого в часі;
   – протокол узгоджується з політиками кожної ланки КФС, враховуючи наявні ресурси.

На рис. 4.7 представлено загальну схему етапів захисту інформації в процесі передачі у КФС від імпланта до лікаря.

Рисунок 4.7 – Наскрізний захист даних у медичній КФС: автентифікація, шифрування, оновлення, сесія

Цей рівень деталізації та адаптивності у виборі криптографічних операцій, а також їхнє розмежування відповідно до можливостей вузлів системи, дозволяє досягти високої ефективності навіть в умовах обмежених ресурсів і нестабільного середовища зв’язку.

Інформаційна взаємодія в межах медичної кіберфізичної системи (КФС) реалізована як послідовний процес передачі, обробки та захисту даних на кожному з рівнів системи – від сенсорного пристрою до кінцевого користувача (лікаря). Потік даних структуровано з урахуванням наскрізного впровадження криптографічних механізмів, які відповідають функціональним можливостям кожного вузла.

На рівні імпланта формується первинний блок даних із фізіологічними показниками, до якого додається автентифікаційний токен, згенерований на основі часової мітки та апаратного ідентифікатора. Дані шифруються за допомогою полегшеного симетричного алгоритму та передаються на шлюз.

Граничний шлюз приймає зашифровані пакети, здійснює перевірку достовірності токену, виконує повторне шифрування згідно з політиками безпеки або делегує частину криптографічних операцій у хмарне середовище. У разі порушення з’єднання з хмарою застосовується механізм буферизації та повторної передачі після відновлення каналу.

У хмарному середовищі здійснюється дешифрування, додаткове шифрування для збереження, логування подій доступу та реалізація політик контролю доступу на основі сертифікатів. Запроваджено механізм повторного використання сесійних ключів у межах дозволеного часовго інтервалу з метою зменшення затримок під час втрати з’єднання.

Застосунок лікаря, після проходження багатофакторної автентифікації, отримує доступ до зашифрованих даних із хмарного сервера через захищене з’єднання. Відображення даних здійснюється лише в активній сесії з обмеженим часом дії, що запобігає несанкціонованому доступу.

На рисунку 4.8 представлено послідовність обробки даних та застосування захисних механізмів на кожному етапі маршруту – від моменту формування до візуалізації.

Рисунок 4.8 – Потік даних і захисних механізмів у медичній КФС: від імпланта до інтерфейсу лікаря

Як видно з рис. 4.8 потік починається з імпланта, який генерує фізіологічні дані пацієнта, формує токен автентифікації на основі часової мітки та виконує первинне шифрування даних. Далі ці дані передаються через безпечне BLE-з'єднання до шлюзу, що функціонує під керуванням FreeRTOS, де здійснюється перевірка автентичності та, за потреби, повторне шифрування. У хмарному середовищі реалізовано механізми контролю доступу, збереження та сесійної взаємодії, включно з делегуванням криптографічних обчислень. Фінальним

етапом є відображення зашифрованих даних у застосунку лікаря через захищене HTTPS-з’єднання з мультифакторною автентифікацією. Кожна ланка включає захисні модулі, адаптовані до її обчислювальних можливостей, що забезпечує наскрізну конфіденційність, цілісність і безперервність сесії.

4.3.2. Експериментальна оцінка продуктивності, енергоефективності та криптостійкості

Метод було оцінено за критеріями, що наведені у пункті 4.3, шляхом проведення низки експериментів на апаратних та емуляційних платформах. Усі експерименти проводились у репрезентативних умовах функціонування медичних кіберфізичних систем. Під час оцінювання враховано обмеження за обсягом оперативної пам’яті, рівнем споживання енергії та швидкодією процесора, що є типовими для вбудованих пристроїв.

У порівнянні брали участь запропонований метод побудови захищеної сесії, класичний протокол DTLS на основі OpenSSL, а також полегшений протокол EDHOC. Реалізації тестувались у середовищах FreeRTOS та bare-metal на мікроконтролерах STM32F407VG та ESP32-S3.

Для оцінки продуктивності було виміряно час виконання процедури встановлення сесії, включаючи генерацію ключів, автентифікацію, та обмін криптографічними параметрами. Запропонований метод продемонстрував середнє значення 12 мс, що майже удвічі менше порівняно з EDHOC (21 мс) та більше ніж удвічі менше порівняно з DTLS (27 мс). Скорочення часу забезпечується завдяки зменшенню кількості кроків обміну та оптимізації структур токенів. На рисунку 4.9 ппредставлено графік часу встановлення сесії для кожного з методів.

На графіку продемонстровано зменшення часу встановлення сесії у запропонованому методі порівняно з EDHOC та DTLS. Перевага особливо помітна при збільшенні кількості паралельних вузлів.

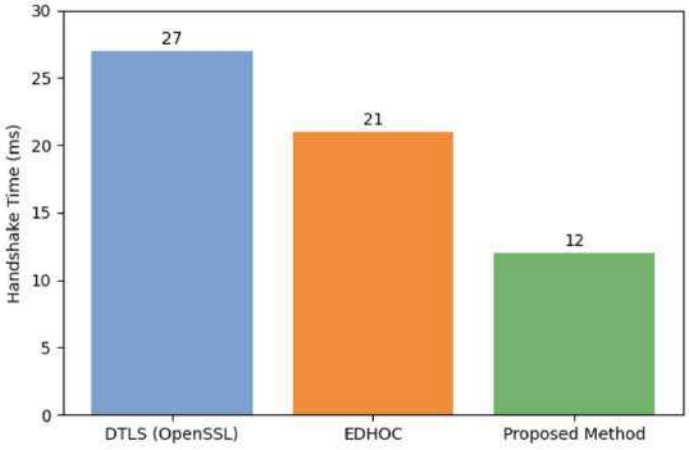

Рисунок 4.9 – Час встановлення захищеної сесії для різних методів

Наступним критерієм стала енергоефективність, яка оцінювалась за допомогою вимірювання спожитої енергії під час криптографічного сеансу. Результати свідчать, що запропонований метод споживає на 37 % менше енергії порівняно з EDHOC і на 54 % менше порівняно з DTLS.

На рисунку 4.10 наведено результати порівняння енергоспоживання.

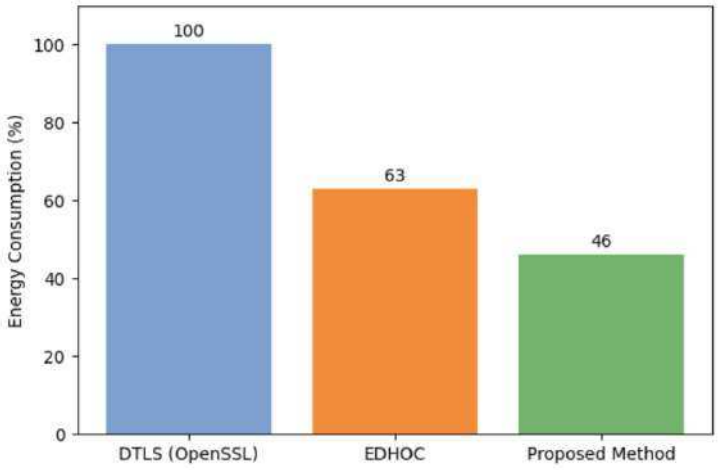

Рисунок 4.10 – Енергоспоживання під час встановлення захищеної сесії

З графіка видно, що запропонований метод характеризується найнижчим рівнем енергоспоживання, що є критично важливим для автономних сенсорних вузлів з обмеженим джерелом живлення.

Оцінка використання оперативної пам’яті продемонструвала, що запропонований метод потребує від 4 до 8 КБ RAM, тоді як EDHOC – понад 20

КБ, а DTLS – понад 32 КБ. Це дає змогу реалізовувати захищену взаємодію навіть на пристроях ультранизького класу.

Результати використання оперативної пам’яті показано на рисунку 4.11.

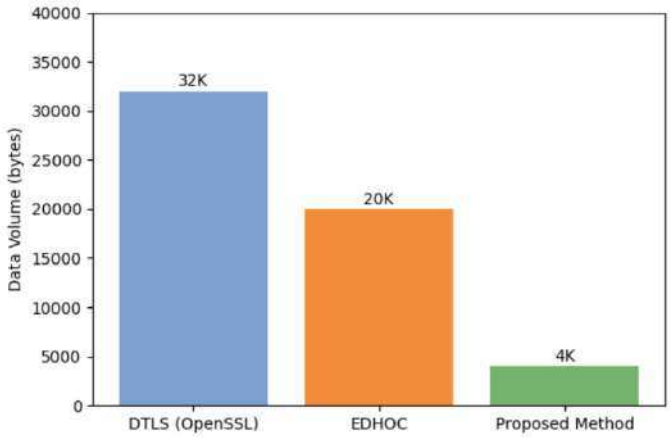

Рисунок 4.11 – Порівняння використання оперативної пам’яті

Запропонований метод має найменші вимоги до обсягу пам’яті, що підтверджує його придатність для bare-metal реалізацій на мікроконтролерах з мінімальними ресурсами.

Криптостійкість аналізувалась за стійкістю до атак типу replay, підміни даних та DoS. Передача даних супроводжується часовими мітками та одноразовими nonce, що унеможливлює повторне відтворення застарілих повідомлень. Перевірка цілісності здійснюється за допомогою MAC-контролю на кожному етапі обміну, а скорочений цикл встановлення сесії зменшує ризики виведення з ладу шляхом перевантаження.

Додатково проведено аналіз масштабованості, що підтвердив лінійне зростання часу ініціалізації при додаванні нових вузлів зі стабільним коефіцієнтом масштабування 1,12 мс/вузол. У той час як DTLS демонстрував експоненційне зростання при перевищенні 25 активних вузлів у кластері.

Інтеграція запропонованого методу у FreeRTOS та bare-metal середовища не потребувала сторонніх бібліотек і обійшлась без використання динамічної пам’яті, що забезпечує передбачуваність виконання та реальну сумісність з вимогами до медичних КФС.

Узагальнена порівняльна оцінка подана в таблиці 4.5.

Таблиця 4.5 – Порівняння параметрів захищених методів комунікації в КФС

Показник	DTLS (OpenSSL)	EDHOC	Запропонований метод
Час встановлення сесії, мс	27	21	12
Енергоспоживання, умовні одиниці	1,00	0,84	0,63
Обсяг RAM, КБ	≥32	≥20	≥4
Кількість раундів встановлення	4	3	2
Витривалість до змін топології	Середня	Висока	Висока
Сумісність з bare-metal	Обмежена	Часткова	Повна

Результати експериментального оцінювання демонструють відповідність запропонованого методу сучасним вимогам до захищеності, продуктивності й енергоефективності в умовах обмежених ресурсів та гетерогенних середовищ.

4.4. Висновки до розділу 4

Набув подальшого розвитку метод побудови комплексної системи криптографічного захисту інформації в кіберфізичних системах, що базується на інтеграції механізмів захищеного завантаження, адаптивного вибору полегшених криптографічних алгоритмів і організації захищеної сесійної взаємодії між компонентами. Підхід охоплює всі рівні семирівневої архітектури КФС та забезпечує узгоджене функціонування криптографічних механізмів упродовж повного життєвого циклу обробки даних – від сенсорного до прикладного рівня.

Розроблено архітектуру комплексного захисту, що передбачає послідовне формування довірчого середовища на основі Secure Boot і кореня довіри, узгодження сесійних ключів із використанням часових токенів і розподіл криптографічного навантаження між вбудованими та периферійними вузлами.

На кожному рівні здійснюється вибір механізмів з урахуванням обчислювальних ресурсів, енергетичних обмежень і характеристик каналів зв’язку, що зменшує навантаження на сенсорні пристрої та підвищує ефективність взаємодії в гетерогенному середовищі.

Запропоновано алгоритм реалізації методу, який забезпечує ініціалізацію довірчого стану, встановлення захищених сесій, динамічне узгодження ключів і адаптивний розподіл криптографічних операцій залежно від стану системи. Передбачено перемикання між локальним і делегованим виконанням, що підтримує стабільність функціонування КФС при зміні топології та умов зв’язку.

Створено імітаційну модель реалізації методу в гетерогенному середовищі з використанням мікроконтролерів STM32 та ESP32, ОС реального часу FreeRTOS і середовища Renode. Модель відтворює повний цикл обробки даних у медичній КФС, включаючи захищене зчитування, передачу, обробку та візуалізацію, а також дозволяє оцінити поведінку системи в умовах обмежених ресурсів і нестабільного зв’язку.

Експериментальне дослідження показало зниження обчислювального навантаження на 35–40% за рахунок адаптивного розподілу криптографічних операцій і використання полегшених алгоритмів, а також підвищення продуктивності системи на 20–25% порівняно з типовими реалізаціями. Додатково підтверджено скорочення часу встановлення сесії, зниження енергоспоживання та вимог до оперативної пам’яті, що підтверджує придатність методу для вбудованих пристроїв.

Отримані результати підтверджують, що запропонований метод забезпечує узгоджену інтеграцію криптографічних механізмів у КФС, ефективне використання ресурсів і підвищення стійкості до типових атак зі збереженням цілісності, автентичності та конфіденційності даних у динамічному середовищі.

ВИСНОВКИ

У дисертаційній роботі вирішено актуальну науково-технічну задачу, що полягає у підвищенні ефективності криптографічного захисту інформації у компонентах кіберфізичних систем, функціонування яких відбувається в умовах обмежених обчислювальних та енергетичних ресурсів. Актуальність дослідження зумовлена зростанням кількості атак на вбудовані пристрої, сенсорні мережі та критичні інфраструктури, а також необхідністю адаптації криптографічних механізмів до реальних умов експлуатації. У роботі запропоновано комплексний підхід, що поєднує моделі захищеного завантаження, безпечного інформаційного обміну та адаптивного розподілу криптографічного навантаження, що дозволяє узгодити вимоги до криптостійкості, продуктивності та енергоефективності у середовищах типу edge/fog.

Найбільш значущі результати роботи полягають у наступному:

1. Вперше побудовано модель криптографічного захисту інформаційного обміну в середовищі компонентів кіберфізичних систем, яка враховує децентралізовану взаємодію вузлів, змінну топологію мережі та обмеження ресурсів. Запропоновані полегшені протоколи встановлення захищених каналів і механізми захисту від атак повторної передачі забезпечують стабільну роботу мережі з 20 вузлів із динамічною маршрутизацією, зменшуючи затримку передавання до 1,4 мс. Це дозволяє підтримувати цілісність і конфіденційність даних у реальному часі навіть в умовах нестабільного з’єднання та змінної структури мережі, що підтверджує ефективність моделі для edge/fog-архітектур.

2. Удосконалено модель захисту інформації на рівні сенсорних модулів та вбудованих пристроїв, яка, на відміну від існуючих підходів, інтегрує механізми Secure Boot, контроль цілісності програмного коду та управління ключовим матеріалом з урахуванням обмежених ресурсів. Реалізація моделі на мікроконтролерах STM32 та ESP32 показала, що перевірка автентичності

прошивки обсягом до 128 КБ виконується за час до 120 мс із використанням не більше 5 КБ оперативної пам’яті, що забезпечує мінімальний вплив на процес ініціалізації пристрою. При цьому досягається суттєве підвищення стійкості до атак підміни прошивки та ін’єкції шкідливого коду, що особливо критично для систем із фізичним доступом до пристроїв.

3. Удосконалено модель динамічного розподілу криптографічного навантаження між компонентами КФС та периферійними обчисленнями, яка базується на узагальненій функції вартості з урахуванням енергоспоживання, затримок і доступних ресурсів. Практична реалізація моделі на платформі STM32F407 дозволила знизити енергоспоживання криптографічних операцій у середньому на 23% порівняно з використанням базового алгоритму AES-128. Крім того, забезпечено оптимізацію часу виконання криптографічних операцій за рахунок адаптивного вибору між локальним виконанням і делегуванням обчислень на edge-рівень, що дозволяє ефективно використовувати ресурси гетерогенних систем.

4. Розвинуто метод побудови комплексної системи криптографічного захисту інформації у компонентах КФС, який інтегрує механізми захищеного завантаження, адаптивного вибору криптографічних алгоритмів та безпечного обміну даними у єдину узгоджену систему. Імітаційне та програмно-апаратне моделювання (STM32, ESP32, FreeRTOS, Renode) показало, що застосування методу дозволяє знизити обчислювальне навантаження на 35–40% та підвищити загальну продуктивність системи на 20–25% у порівнянні з традиційними підходами. Це підтверджує доцільність використання запропонованого методу для побудови ефективних систем криптографічного захисту в умовах обмежених ресурсів.

СПИСОК ДЖЕРЕЛ

[1] Zhang K., Shi Y., Karnouskos S., Sauter T., Fang H., Colombo A. W. Advancements in industrial cyber-physical systems: An overview and perspectives. *IEEE Transactions on Industrial Informatics.* 2022. Vol. 19, no. 1. P. 716–729. URL: https://doi.org/10.1109/TII.2022.3199481

[2] Shamsuzzaman H. M., Mosleuzzaman M. D., Paran M. S. Real-time cybersecurity integration in PLCs and IoT gateways: A performance-conscious approach to encryption, authentication, and intrusion detection. *American Journal of Advanced Technology and Engineering Solutions.* 2025. Vol. 1, no. 2. P. 32–57. URL: https://doi.org/10.63125/g937f918

[3] Sakthivel V., Rao V., Prakash P., Lee J. W. Enhancing Internet of Things security through quantum cryptography. *Quantum Computing, Cyber Security and Cryptography: Issues, Technologies, Algorithms, Programming and Strategies.* Singapore: Springer Nature Singapore, 2025. P. 323–352. URL: https://doi.org/10.1007/978-981-96-4948-8_13

[4] Zhang S., Yang Y., He J., Liang W., Li K., Crespo R. G. FKSS: A fast keyword search scheme with access control for cloud-assisted edge computing. *International Journal of Information Security.* 2026. Vol. 25, no. 1. P. 21. URL: https://doi.org/10.1007/s10207-025-01175-0

[5] Tanveer M., Alharbi A. G., Akhtar M. J. A secure and resource-efficient authenticated key agreement framework for mobile edge computing. *Peer-to-Peer Networking and Applications.* 2025. Vol. 18, no. 4. P. 200. URL: https://doi.org/10.1007/s12083-025-02016-6

[6] Ahmad S., Mehfuz S., Urooj S., Alsubaie N. Machine learning-based intelligent security framework for secure cloud key management. *Cluster Computing.* 2024. Vol. 27, no. 5. P. 5953–5979. URL: https://doi.org/10.1007/s10586-024-04288-8

[7] Celiktas B., Celikbilek I., Ozdemir E. A higher-level security scheme for key access on cloud computing. *IEEE Access.* 2021. Vol. 9. P. 107347–107359. URL: https://doi.org/10.1109/ACCESS.2021.3101048

[8] Singh S., Sharma P. K., Moon S. Y., Park J. H. Advanced lightweight encryption algorithms for IoT devices: Survey, challenges and solutions. *Journal of Ambient Intelligence and Humanized Computing.* 2024. Vol. 15, no. 2. P. 1625–1642. URL: https://doi.org/10.1007/s12652-017-0494-4

[9] Khalifa M., Algarni F., Khan M. A., Ullah A., Aloufi K. A lightweight cryptography framework to secure memory heap in Internet of Things. *Alexandria Engineering Journal.* 2021. Vol. 60, no. 1. P. 1489–1497. URL: https://doi.org/10.1016/j.aej.2020.11.003

[10] Черненко Р. Оцінка продуктивності алгоритмів легкої криптографії на обмежених 8-бітних пристроях. *Кібербезпека: освіта, наука, техніка.* 2023. № 1(21). С. 273–285. URL: https://doi.org/10.28925/2663-4023.2023.21.273285

[11] Шкітов А., Авдалов Г. Оцінювання продуктивності гібридних криптографічних схем при впровадженні у системи з обмеженими ресурсами в умовах квантової загрози. *Measuring and Computing Devices in Technological Processes.* 2025. № 4. С. 316–322. URL: https://doi.org/10.31891/2219-9365-2025-84-36

[12] Hu C. L., Wang L., Chen M. L., Pei C. A real-time interactive decision-making and control framework for complex cyber-physical-human systems. *Annual Reviews in Control.* 2024. Vol. 57. P. 100938. URL: https://doi.org/10.1016/j.arcontrol.2024.100938

[13] Peng Y., Jolfaei A., Hua Q., Shang W. L., Yu K. Real-time transmission optimization for edge computing in industrial cyber-physical systems. *IEEE Transactions on Industrial Informatics.* 2022. Vol. 18, no. 12. P. 9292–9301. URL: https://doi.org/10.1109/TII.2022.3181199

[14] Korki M., Jin J., Tian Y. C. Real-time cyber-physical systems: State-of-the-art and future trends. *Handbook of Real-Time Computing. Singapore: Springer Nature Singapore,* 2022. P. 509–540. URL: https://doi.org/10.1007/978-981-287-251-7_37

[15] Sukiasyan A., Badikyan H., Pedrosa T., Leitao P. Secure data exchange in Industrial Internet of Things. *Neurocomputing.* 2022. Vol. 484. P. 183–195. URL: https://doi.org/10.1016/j.neucom.2021.07.101

[16] Lara-Nino C. A., Diaz-Perez A., Morales-Sandoval M. Key-establishment protocols for constrained cyber-physical systems. *Security in Cyber-Physical Systems: Foundations and Applications. Cham: Springer,* 2021. P. 39–65. URL: https://doi.org/10.1007/978-3-030-67361-1_2

[17] Verma R. Smart city healthcare cyber physical system: Characteristics, technologies and challenges. *Wireless Personal Communications.* 2022. Vol. 122, no. 2. P. 1413–1433. URL: https://doi.org/10.1007/s11277-021-08955-6

[18] Cieślak E. Unmanned aircraft systems. *Safety & Defense.* 2021. Vol. 7, no. 1. P. 72–82. URL: https://doi.org/10.37105/sd.110

[19] Cosar M. Cyber attacks on unmanned aerial vehicles and cyber security measures. *The Eurasia Proceedings of Science Technology Engineering and Mathematics.* 2022. Vol. 21. P. 258–265. URL: https://dergipark.org.tr/en/download/article-file/2860741

[20] Oruc A. Potential cyber threats, vulnerabilities, and protections of unmanned vehicles. *Drone Systems and Applications.* 2022. Vol. 10, no. 1. P. 51–58. URL: https://cdnsciencepub.com/doi/pdf/10.1139/juvs-2021-0022

[21] Mejía-Granda C. M., Fernández-Alemán J. L., Carrillo-de-Gea J. M., García-Berná J. A. Security vulnerabilities in healthcare: An analysis of medical devices and software. *Medical & Biological Engineering & Computing.* 2024. Vol. 62, no. 1. P. 257–273. URL: https://doi.org/10.1007/s11517-023-02912-0

[22] Newaz A. I., Sikder A. K., Rahman M. A., Uluagac A. S. A survey on security and privacy issues in modern healthcare systems: Attacks and defenses. *ACM Transactions on Computing for Healthcare.* 2021. Vol. 2, no. 3. P. 1–44. URL: https://dl.acm.org/doi/abs/10.1145/3453176

[23] Hasan M. K., Ghazal T. M., Saeed R. A., Pandey B., Gohel H., Eshmawi A. A., Alkhassawneh H. M. A review on security threats, vulnerabilities, and counter measures of 5G enabled Internet-of-Medical-Things. *IET Communications.* 2022. Vol. 16, no. 5. P. 421–432. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/cmu2.12301

[24] Abraham D., Houmb S. H., Erdodi L. Cyber-attacks on energy infrastructure – A literature overview and perspectives on the current situation. *Applied Sciences.* 2025. Vol. 15, no. 17. P. 9233. URL: https://doi.org/10.3390/app15179233

[25] Tatipatri N., Arun S. L. A comprehensive review on cyber-attacks in power systems: Impact analysis, detection, and cyber security. *IEEE Access.* 2024. Vol. 12. P. 18147–18167. URL: https://doi.org/10.1109/ACCESS.2024.3361039

[26] Rajkumar V. S., Ştefanov A., Presekal A., Palensky P., Torres J. L. R. Cyber attacks on power grids: Causes and propagation of cascading failures. *IEEE Access.* 2023. Vol. 11. P. 103154–103176. URL: https://doi.org/10.1109/ACCESS.2023.3317695

[27] Boubiche D. E., Athmani S., Boubiche S., Toral-Cruz H. Cybersecurity issues in wireless sensor networks: Current challenges and solutions. *Wireless Personal Communications.* 2021. Vol. 117, no. 1. P. 177–213. URL: https://doi.org/10.1007/s11277-020-07213-5

[28] Khashan O. A., Ahmad R., Khafajah N. M. An automated lightweight encryption scheme for secure and energy-efficient communication in wireless sensor networks. *Ad Hoc Networks.* 2021. Vol. 115. P. 102448. URL: https://doi.org/10.1016/j.adhoc.2021.102448

[29] Sharmila, Kumar P., Bhushan S., Kumar M., Alazab M. Secure key management and mutual authentication protocol for wireless sensor network by linking edge devices using hybrid approach. *Wireless Personal Communications*. 2023. Vol. 130, no. 4. P. 2935–2957. URL: https://doi.org/10.1007/s11277-023-10410-7

[30] Gautam A. K., Kumar R. A comprehensive study on key management, authentication and trust management techniques in wireless sensor networks. *SN Applied Sciences.* 2021. Vol. 3, no. 1. P. 50. URL: https://doi.org/10.1007/s42452-020-04089-9

[31] Wang Q., Li H. Application of IoT authentication key management algorithm to personnel information management. *Computational Intelligence and Neuroscience*. 2022. Vol. 2022, no. 1. P. 4584072. URL: https://onlinelibrary.wiley.com/doi/abs/10.1155/2022/4584072

[32] Babatope J. Designing energy-efficient cryptographic modules for securing real-time data communication in embedded Internet of Things (IoT) devices. *International Research Journal of Modernization in Engineering, Technology and Science*. 2025. Vol. 7, no. 8. P. 1995. URL: https://doi.org/10.56726/IRJMETS81995

[33] Zhang Z., Wang X., Hao Q., Xu D., Zhang J., Liu J., Ma J. High-efficiency parallel cryptographic accelerator for real-time guaranteeing dynamic data security in embedded systems. *Micromachines*. 2021. Vol. 12, no. 5. P. 560. URL: https://doi.org/10.3390/mi12050560

[34] Silva C., Cunha V. A., Barraca J. P., Aguiar R. L. Analysis of the cryptographic algorithms in IoT communications. *Information Systems Frontiers*. 2024. Vol. 26, no. 4. P. 1243–1260. URL: https://doi.org/10.1007/s10796-023-10383-9

[35] Hadjadj M. A., Sadoudi S., Azzaz M. S., Bendecheche H., Kaibou R. A new hardware architecture of lightweight and efficient real-time video chaos-based encryption algorithm. *Journal of Real-Time Image Processing*. 2022. Vol. 19, no. 6. P. 1049–1062. URL: https://doi.org/10.1007/s11554-022-01244-w

[36] El-Hajj M., Mousawi H., Fadlallah A. Analysis of lightweight cryptographic algorithms on IoT hardware platform. *Future Internet*. 2023. Vol. 15, no. 2. P. 54. URL: https://doi.org/10.3390/fi15020054

[37] Suryateja P. S., Rao K. V. A survey on lightweight cryptographic algorithms in IoT. *Cybernetics and Information Technologies*. 2024. Vol. 24, no. 1. P. 21–34. URL: https://doi.org/10.2478/cait-2024-0002

[38] Thakor V. A., Razzaque M. A., Khandaker M. R. Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison and research opportunities. *IEEE Access*. 2021. Vol. 9. P. 28177–28193. URL: https://doi.org/10.1109/ACCESS.2021.3052867

[39] Sabri O., Al-Shargabi B., Abuarqoub A., Hakami T. A. A lightweight encryption method for IoT-based healthcare applications: A review and future prospects. *IoT*. 2025. Vol. 6, no. 2. P. 23. URL: https://doi.org/10.3390/iot6020023

[40] Chen Y., Bao Z., Yu H. Differential-linear approximation semi-unconstrained searching and partition tree: Application to LEA and SPECK. *International Conference on the Theory and Application of Cryptology and Information Security*. Singapore: Springer Nature Singapore, 2023. P. 223–255. URL: https://doi.org/10.1007/978-981-99-8727-6_8

[41] Sleem L., Couturier R. Speck-R: An ultra light-weight cryptographic scheme for Internet of Things. *Multimedia Tools and Applications*. 2021. Vol. 80, no. 11. P. 17067–17102. URL: https://doi.org/10.1007/s11042-020-09625-8

[42] Khalil A., Hamarsheh M. M., Salah S. MPRESENT: A lightweight cryptographic cipher for optimizing energy efficiency in IoT applications. *Arabian Journal for Science and Engineering*. 2025. P. 1–13. URL: https://doi.org/10.1007/s13369-025-10432-2

[43] Alshawish I., Al-Haj A. An efficient mutual authentication scheme for IoT systems. *The Journal of Supercomputing*. 2022. Vol. 78, no. 14. P. 16056–16087. URL: https://doi.org/10.1007/s11227-022-04520-5

[44] Allakany A., Saber A., Mostafa S. M., Alsabaan M., Ibrahem M. I., Elwahsh H. Enhancing security in ZigBee wireless sensor networks: A new approach and mutual authentication scheme for D2D communication. *Sensors*. 2023. Vol. 23, no. 12. P. 5703. URL: https://doi.org/10.3390/s23125703

[45] Singh S., Chaurasiya V. K. Mutual authentication scheme of IoT devices in fog computing environment. *Cluster Computing*. 2021. Vol. 24, no. 3. P. 1643–1657. URL: https://doi.org/10.1007/s10586-020-03211-1

[46] Vinoth R., Deborah L. J. An efficient key agreement and authentication protocol for secure communication in industrial IoT applications. *Journal of Ambient Intelligence and Humanized Computing*. 2023. Vol. 14, no. 3. P. 1431–1443. URL: https://doi.org/10.1007/s12652-021-03167-z

[47] Zhang Y., Cheng K., Khan F., Alturki R., Khan R., Rehman A. U. A mutual authentication scheme for establishing secure device-to-device communication sessions in the edge-enabled smart cities. *Journal of Information Security and Applications*. 2021. Vol. 58. P. 102683. URL: https://doi.org/10.1016/j.jisa.2020.102683

[48] Марченко Р., Коваленко А., Знайдюк В. Аналіз методів виявлення аномального трафіку в мережах IoT. *Системи управління, навігації та зв'язку*. 2024. № 1(75). С. 133–136. URL: https://doi.org/10.26906/SUNZ.2024.1.133

[49] Андрущак І., Кошелюк В. Окремі аспекти проектування системи інформаційної безпеки для захисту IoT-мереж від атак. *International Science Journal of Engineering & Agriculture*. 2025. № 4(5). С. 27–39. URL: https://doi.org/10.46299/j.isjea.20250405.03

[50] Петрушко І., Поліщук В., Матей А. Постквантова криптографія: виклики та перспективи розробки квантово-стійких алгоритмів для забезпечення безпеки інформаційних систем. *Herald of Khmelnytskyi National University. Technical Sciences*. 2025. № 347(1). С. 461–467. URL: https://doi.org/10.31891/2307-5732-2025-347-63

[51] Макаренко А. О., Жураковський Б. Ю., Осипчук С. О., Григоренко О. Г., Лемешко А. В. Методи побудови захищених комунікаційних каналів для IoT-пристроїв у мережах п’ятого покоління. *Наукові записки Державного університету інформаційно-комунікаційних технологій*. 2025. № 2. С. 183–193.

[52] Zhang K., Shi Y., Karnouskos S., Sauter T., Fang H., Colombo A. W. Advancements in industrial cyber-physical systems: An overview and perspectives. *IEEE Transactions on Industrial Informatics*. 2022. Vol. 19, no. 1. P. 716–729. URL: https://doi.org/10.1109/TII.2022.3199481

[53] Rozlomii I., Yarmilko A., Naumenko S. Security and efficiency models for cyber-physical systems in medical devices. *2024 IEEE 19th International Conference on Computer Science and Information Technologies (CSIT)*. 2024. P. 1–4. URL: https://doi.org/10.1109/CSIT65290.2024.10982678

[54] Khujamatov H., Reypnazarov E., Khasanov D., Akhmedov N. IoT, IIoT, and cyber-physical systems integration. *Emergence of Cyber Physical System and IoT in Smart Automation and Robotics*. Cham: Springer, 2021. P. 31–50. URL: https://doi.org/10.1007/978-3-030-66222-6_3

[55] Sobb T., Turnbull B., Moustafa N. A holistic review of cyber–physical–social systems: New directions and opportunities. *Sensors*. 2023. Vol. 23, no. 17. P. 7391. URL: https://doi.org/10.3390/s23177391

[56] Yarmilko A., Rozlomii I., Naumenko S. Dependability of embedded systems in the Industrial Internet of Things: Information security and reliability of the communication cluster. *Information Technology for Education, Science, and Technics*. 2024. Vol. 222. P. 235–249. URL: https://doi.org/10.1007/978-3-031-71804-5_16

[57] Vlastos P., Sljivo I., Carter C., Woodard A. Developing a dependable multi-agent rover swarm using cFS. *2024 IEEE 35th International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2024. P. 377–382. URL: https://doi.org/10.1109/ISSREW63542.2024.00105

[58] Pham L. N. H. Exploring cyber-physical energy and power system: Concepts, applications, challenges, and simulation approaches. *Energies*. 2022. Vol. 16, no. 1. P. 42. URL: https://doi.org/10.3390/en16010042

[59] Chowdhury R. H., Mostafa B. Cyber-physical systems for critical infrastructure protection: developing advanced systems to secure energy grids, transportation networks, and water systems from cyber threats. *Journal of Computer Science and Electrical Engineering*. 2025. Vol. 7, no. 1. P. 16–26. URL: https://doi.org/10.61784/jcsee3027

[60] Розломій І. О., Ярмілко А. В., Науменко С. В. Інтелектуальні підходи до організації інформаційного обміну в динамічних зграях безпілотних платформ. *Штучний інтелект та інтелектуальні системи:* матеріали XXIV Міжнародної науково-технічної конференції AIIS'2024. 2024. С. 144–149.

[61] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. IoT Smart Implants: Information security and the implementation of lightweight cryptography. *Proceedings of the 6th International Conference on Informatics & Data-Driven Medicine (IDDM'2023)*. 2023. P. 145–146. URL: https://ceur-ws.org/Vol-3609/paper12.pdf

[62] Розломій І. О., Косенюк Г. В., Науменко С. В., Михайловський П. В. Моделювання системи датчиків на базі мікроконтролера в ігровій симуляції «Смарт-будинок» з використанням шифрування. *Computer-Integrated Technologies: Education, Science, Production*. 2023. № 53. С. 292–299. URL: https://doi.org/10.36910/6775-2524-0560-2023-53-43

[63] Jeffrey N., Tan Q., Villar J. R. A review of anomaly detection strategies to detect threats to cyber-physical systems. *Electronics*. 2023. Vol. 12, no. 15. P. 3283. URL: https://doi.org/10.3390/electronics12153283

[64] Bajic B., Rikalovic A., Suzic N., Piuri V. Toward a human-cyber-physical system for real-time anomaly detection. *IEEE Systems Journal*. 2024. Vol. 18, no. 2. P. 1308–1319. URL: https://doi.org/10.1109/JSYST.2024.3402978

[65] Ain Q. U., Jilani A. A. A., Butt N. A., Rehman S. U., Alhulayyil H. A. Anomaly detection for aviation cyber-physical system: Opportunities and challenges. *IEEE Access*. 2024. Vol. 12. P. 175905–175925. URL: https://doi.org/10.1109/ACCESS.2024.3495519

[66] Rozlomii I., Yarmilko A., Naumenko S. Data security of IoT devices with limited resources: Challenges and potential solutions. *Proceedings of the 4th Edge Computing Workshop (DOORS 2024)*. 2024. Vol. 3666. P. 85–96. URL: https://ceur-ws.org/Vol-3666/paper13.pdf

[67] Zhang K., Shi Y., Karnouskos S., Sauter T., Fang H., Colombo A. W. Advancements in industrial cyber-physical systems: An overview and perspectives. *IEEE Transactions on Industrial Informatics*. 2022. Vol. 19, no. 1. P. 716–729. URL: https://doi.org/10.1109/TII.2022.3199481

[68] Serôdio C., Mestre P., Cabral J., Gomes M., Branco F. Software and architecture orchestration for process control in industry 4.0 enabled by cyber-physical systems technologies. *Applied Sciences*. 2024. Vol. 14, no. 5. P. 2160. URL: https://doi.org/10.3390/app14052160

[69] Nounou A., Jaber H., Aydin R. A cyber-physical system architecture based on lean principles for managing Industry 4.0 setups. *International Journal of Computer Integrated Manufacturing*. 2022. Vol. 35, no. 8. P. 890–908. URL: https://doi.org/10.1080/0951192X.2022.2027016

[70] Mustafa R., Sarkar N. I., Mohaghegh M., Pervez S., Morados R. A secure and energy-efficient cross-layer network architecture for the Internet of Things. *Sensors*. 2025. Vol. 25, no. 11. P. 3457. URL: https://doi.org/10.3390/s25113457

[71] Faure E., Rozlomii I., Naumenko S. Cryptographic load sharing method in critical infrastructure sensor networks. *Proceedings of the Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN-25).* 2025. P. 24–34. URL: https://ceur-ws.org/Vol-4024/paper01.pdf

[72] Rahim R. Secure and energy-optimized VLSI architecture for edge-enabled embedded systems. *Journal of VLSI and Embedded System Design*. 2025. P. 41–48. URL: https://iaeces.com/Index/index.php/JVESD/article/view/30

[73] Song X. Microcontroller hardware design and signal processing optimization. *Journal of Computing and Electronic Information Management*. 2025. Vol. 17, no. 2. P. 42–46. URL: https://doi.org/10.54097/hddrb865

[74] Ma Y., Wang Y., Di Cairano S., Koike-Akino T., Guo J., Orlik P., Lu C. Smart actuation for end-edge industrial control systems. *IEEE Transactions on Automation Science and Engineering*. 2022. Vol. 21, no. 1. P. 269–283. URL: https://doi.org/10.1109/TASE.2022.3216217

[75] Apsite I., Salehi S., Ionov L. Materials for smart soft actuator systems. *Chemical Reviews*. 2021. Vol. 122, no. 1. P. 1349–1415. URL: https://doi.org/10.1021/acs.chemrev.1c00453

[76] Розломій І., Ярмілко А., Науменко С. Ресурсоощадний підхід до побудови secure boot у вбудованих системах інтернету речей. *Системи контролю інформації та інтелектуальні технології. Досягнення та застосування*. Львів–Торунь : Liha-Pres, 2025. С. 102–122. URL: https://doi.org/10.36059/978-966-397-538-2-6

[77] Аронов А. О. Дослідження методик оптимізації параметрів системи керування розумним будинком з використанням IoT. *Телекомунікаційні та інформаційні технології*. 2025. № 1. С. 141–150. URL: https://doi.org/10.31673/2412-4338.2025.013027

[78] Ровінський В. А., Євчук О. В., Ровінський Ю. В. Вибір операційних систем реального часу при розробці пристроїв для технічної діагностики. *Methods and Devices of Quality Control*. 2022. № 1(48). С. 66–77. URL: https://doi.org/10.31471/1993-9981-2022-1(48)-66-77

[79] Dunbar N. SPI interrupt. *Arduino Interrupts: Harness the Power of Interrupts in Your Arduino and ATmega328 Code*. Berkeley: Apress, 2023. P. 95–124. URL: https://doi.org/10.1007/978-1-4842-9714-8_7

[80] Currie E. H. Communication peripherals. *Mixed-Signal Embedded Systems Design: A Hands-on Guide to the Cypress PSoC*. Cham: Springer, 2021. P. 347–434. URL: https://doi.org/10.1007/978-3-030-70312-7_9

[81] Kurapati M. A comparative overview of I2C and I3C protocols for server platform management and diagnostics. *Journal of Computational Analysis & Applications*. 2025. Vol. 34, no. 10. URL: https://doi.org/10.48047/jocaaa.2025.34.10.16

[82] Subero A. USART, SPI, I2C, and communication protocols. *Programming PIC Microcontrollers with XC8: Mastering Classical Embedded Design*. Berkeley: Apress, 2024. P. 297–366. URL: https://doi.org/10.1007/979-8-8688-0467-0_8

[83] Pundir A., Singh S., Kumar M., Bafila A., Saxena G. J. Cyber-physical systems enabled transport networks in smart cities: Challenges and enabling technologies of the new mobility era. *IEEE Access*. 2022. Vol. 10. P. 16350–16364. URL: https://doi.org/10.1109/ACCESS.2022.3147323

[84] Kumari N., Lee K., Ranaweera C. Visually extracting the network topology of drone swarms. *Robotics and Autonomous Systems*. 2025. P. 105313. URL: https://doi.org/10.1016/j.robot.2025.105313

[85] Abdrabou A., Al Darei M. S., Prakash M., Zhuang W. Application-oriented traffic modeling of WiFi-based Internet of Things gateways. *IEEE Internet of Things Journal*. 2021. Vol. 9, no. 2. P. 1159–1170. URL: https://doi.org/10.1109/JIOT.2021.3079115

[86] Katila R., Gia T. N., Westerlund T. Analysis of mobility support approaches for edge-based IoT systems using high data rate Bluetooth Low Energy 5.

Computer Networks. 2022. Vol. 209. P. 108925. URL: https://doi.org/10.1016/j.comnet.2022.108925

[87] Wu W., Wang H., Cheng Z. ReLoRaWAN: Reliable data delivery in LoRaWAN networks with multiple gateways. *Ad Hoc Networks*. 2023. Vol. 147. P. 103203. URL: https://doi.org/10.1016/j.adhoc.2023.103203

[88] Nguyen C. V., Coboi A. E., Bach N. V., Dang A. T., Le T. T., Nguyen H. P., Nguyen M. T. ZigBee based data collection in wireless sensor networks. *International Journal of Informatics and Communication Technology*. 2021. Vol. 10, no. 3. P. 212–224. URL: https://doi.org/10.11591/ijict.v10i3.pp211-224

[89] Höglund R., Tiloca M., Selander G., Mattsson J. P., Vučinić M., Watteyne T. Secure communication for the IoT: EDHOC and (group) OSCORE protocols. *IEEE Access*. 2024. Vol. 12. P. 49865–49877. URL: https://doi.org/10.1109/ACCESS.2024.3384095

[90] Hristozov S., Huber M., Xu L., Fietz J., Liess M., Sigl G. The cost of OSCORE and EDHOC for constrained devices. *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*. 2021. P. 245–250. URL: https://doi.org/10.1145/3422337.3447834

[91] Рзаєва С. Л., Складанний П. М., Костюк Ю. В., Абрамов В. О., Кравченко В. Г. Адаптивне управління інформаційною безпекою в хмарно-орієнтованих інтелектуальних транспортних системах. *Безпека інформації*. 2025. № 31(1). С. 23–36. URL: https://doi.org/10.18372/2225-5036.31.20634

[92] Костюк Ю. В., Складанний П. М., Рзаєва С. Л., Самойленко Ю. О., Коршун Н. В. Інтелектуальні системи керування та захисту в кіберфізичних і хмарних середовищах Smart Grid. *Кібербезпека: освіта, наука, техніка*. 2025. № 2(30). С. 125–156. URL: https://doi.org/10.28925/2663-4023.2025.30.956

[93] Rozlomii I., Yarmilko A., Naumenko S. Innovative resource-saving security strategies for IoT devices. *Journal of Edge Computing*. 2025. Vol. 4, no. 1. P. 35–56. URL: https://doi.org/10.55056/jec.748

[94] Mahato G. K., Chakraborty S. K. Securing edge computing using cryptographic schemes: A review. *Multimedia Tools and Applications*. 2024. Vol. 83, no. 12. P. 34825–34848. URL: https://doi.org/10.1007/s11042-023-15592-7

[95] Valluri B. P., Sharma N. Exceptional key based node validation for secure data transmission using asymmetric cryptography in wireless sensor networks. *Measurement: Sensors*. 2024. Vol. 33. P. 101150. URL: https://doi.org/10.1016/j.measen.2024.101150

[96] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The role of encryption in information protection for cloud computing. *2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST)*. 2024. P. 70–75. URL: https://doi.org/10.1109/SIST61555.2024.10629501

[97] Shermy R. P., Saranya N. Cloud-based big data architecture and infrastructure. *Resilient Community Microgrids*. 2025. P. 131–188. URL: https://doi.org/10.1002/9781394272549.ch6

[98] Sun C., Xing R., Wu Y., Zhou G., Zheng F., Hu D. Design of over-the-air firmware update and management for IoT device with cloud-based RESTful web services. *2021 China Automation Congress (CAC)*. 2021. P. 5081–5085. URL: https://doi.org/10.1109/CAC53003.2021.9727516

[99] Enemosah A., Chukwunweike J. Next-generation SCADA architectures for enhanced field automation and real-time remote control in oil and gas fields. *International Journal of Computer Applications Technology and Research*. 2022. Vol. 11, no. 12. P. 514–529. URL: https://doi.org/10.7753/IJCATR1112.1018

[100] Waqas M., Jamil M. Smart IoT SCADA system for hybrid power monitoring in remote natural gas pipeline control stations. *Electronics*. 2024. Vol. 13, no. 16. P. 3235. URL: https://doi.org/10.3390/electronics13163235

[101] Yu Z., Gao H., Cong X., Wu N., Song H. H. A survey on cyber–physical systems security. *IEEE Internet of Things Journal*. 2023. Vol. 10, no. 24. P. 21670–21686. URL: https://doi.org/10.1109/JIOT.2023.3289625

[102] Tsantikidou K., Sklavos N. Threats, attacks, and cryptography frameworks of cybersecurity in critical infrastructures. *Cryptography*. 2024. Vol. 8, no. 1. P. 7. URL: https://doi.org/10.3390/cryptography8010007

[103] Rozlomii I., Yarmilko A., Naumenko S. Vulnerability modeling in cybersecurity of intelligent infrastructure networks. *International Scientific-Practical Conference*. Cham: Springer Nature Switzerland, 2024. P. 234–248. URL: https://doi.org/10.1007/978-3-031-90735-7_19

[104] Faure E., Rozlomii I., Naumenko S. Hybrid digital twin-driven anomaly detection in IoT telemetry using LSTM autoencoder. *Proceedings of the 2nd International Workshop on Data Analytics (WDA 2026), Kyiv, Ukraine, January 26, 2026.* CEUR Workshop Proceedings, Vol. 4155. 2025. P. 76–89. URL: https://ceur-ws.org/Vol-4155/paper06.pdf

[105] Danchenko O., Rozlomii I., Yarmilko A., Naumenko S. A lightweight Secure Boot mechanism for protecting the firmware of IoT devices. *Proceedings of the 13-th International Conference Information Control Systems & Technologies (ICST 2025), Odesa, Ukraine, September 24–26, 2025. CEUR Workshop Proceedings,* Vol. 4048. 2025. P. 240-250. URL: https://ceur-ws.org/Vol-4048/paper19.pdf

[106] Науменко С. В., Михайловський П. В., Стабецька Т. А. Сучасні технології захисту даних у вбудованих пристроях з обмеженими ресурсами. *Free and Open Source Software: матеріали міжнародної науково-практичної конференції*, Харків, 13–14 лют. 2025 р. Харків, 2025. С. 63–64.

[107] Науменко С. В., Розломій І. О. Information protection strategies in industry 4.0: encryption and cybersecurity for industrial systems. *Theoretical and Experimental Research in Materials Science and Mechanical Engineering:* матеріали IX International scientific and practical conference. Луцьк: Вежа-Друк, 2023. P. 191–193.

[108] Kumar S., Pillai C. S. An analysis of light weight symmetric encryption algorithms for secure data transmission in IoT. *2024 International Conference on*

Intelligent Algorithms for Computational Intelligence Systems (IACIS). 2024. P. 1–4. URL: https://doi.org/10.1109/IACIS61494.2024.10721701

[109] Zabolotnii S., Rozlomii I., Yarmilko A., Naumenko S. Reconfigured CoARX architecture for implementing ARX hashing in microcontrollers of IoT systems with limited resources. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska*. 2025. Vol. 15, no. 4. P. 164–169. URL: https://doi.org/10.35784/iapgos.7782

[110] Panahi P., Bayılmış C., Çavuşoğlu U., Kaçar S. Performance evaluation of lightweight encryption algorithms for IoT-based applications. *Arabian Journal for Science and Engineering*. 2021. Vol. 46, no. 4. P. 4015–4037. URL: https://doi.org/10.1007/s13369-021-05358-4

[111] Rozlomii I., Yarmilko A., Naumenko S. Resource-efficient solutions for data security at the network level of the Medical Internet of Things. *Proceedings of the 7th International Conference on Informatics & Data-Driven Medicine (IDDM'2024)*. 2024. P. 171–182. URL: https://ceur-ws.org/Vol-3892/paper13.pdf

[112] Rozlomii I., Faure E., Yarmilko A., Naumenko S. The method for verifying firmware integrity in IoT devices for secure boot using lightweight hash functions. *Proceedings of the Cyber Security and Data Protection (CSDP 2025), Lviv, Ukraine, July 31, 2025. CEUR Workshop Proceedings,* Vol. 4042. 2025. P. 105-116. URL: https://ceur-ws.org/Vol-4042/paper8.pdf

[113] Rozlomii I. O., Yarmilko A., Naumenko S. Optimized hash functions for integrity control and data recovery in embedded systems. *Materials of the XI International Scientific Conference «Information-Management Systems and Technologies»*. 2023. P. 31–34.

[114] Yu Z., Gao H., Cong X., Wu N., Song H. H. A survey on cyber–physical systems security. *IEEE Internet of Things Journal*. 2023. Vol. 10, no. 24. P. 21670–21686. URL: https://doi.org/10.1109/JIOT.2023.3289625

[115] Kim S., Park K. J., Lu C. A survey on network security for cyber–physical systems: From threats to resilient design. *IEEE Communications Surveys & Tutorials*. 2022. Vol. 24, no. 3. P. 1534–1573. URL: https://doi.org/10.1109/COMST.2022.3187531

[116] Faure E., Rozlomii I., Yarmilko A., Naumenko S. Protection of IoT networks: cryptographic solutions for cybersecurity management. *Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN 2024)*, Kyiv, January 24–27, 2024. 2024. P. 24–34. URL: https://ceur-ws.org/Vol-3925/paper03.pdf

[117] Rozlomii I., Koseniuk G., Naumenko S. Mechanisms for cryptographic code authentication control in sensor nodes with limited computing resources. *Computer-Integrated Technologies: Education, Science, Production*. 2025. № 61. С. 193–198. URL: https://doi.org/10.36910/6775-2524-0560-2025-61-27

[118] Розломій І. О., Науменко С. В., Михайловський П. В. Формування сеансових ключів у сенсорних пристроях з обмеженим обсягом пам’яті на основі часових токенів. *Комплексне забезпечення якості технологічних процесів та систем (КЗЯТПС – 2025):* матеріали XV Міжнародної науково-практичної конференції, Чернігів, 22–23 трав. 2025 р. Чернігів: НУ «Чернігівська політехніка», 2025. Т. 2. С. 248–249.

[119] Rajesh M. A session key based security mechanism for cyber physical system. *Recent Trends in Intensive Computing*. 2021. Vol. 39. P. 245. URL: https://doi.org/10.3233/APC210201

[120] Madushan H., Salam I., Alawatugoda J. A review of the NIST lightweight cryptography finalists and their fault analyses. *Electronics*. 2022. Vol. 11, no. 24. P. 4199. URL: https://doi.org/10.3390/electronics11244199

[121] Rozlomii I., Naumenko S., Mykhailovskyi P., Monarkh V. Resource-saving cryptography for microcontrollers in biomedical devices. *2024 IEEE 5th KhPI*

Week on Advanced Technology (KhPIWeek). 2024. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61434.2024.10877958

[122] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Адаптивна криптографія для енергоефективного захисту пристроїв IoT. *Проблеми моделювання та автоматизації проектування*. 2024. № 1(19). С. 77–83. URL: https://doi.org/10.31474/2074-7888-2024-1-19-77-83

[123] Al-shmailawi H. A., Al-Hemiary E. H., Sikora A. Comprehensive review of NIST lightweight cryptography algorithms for IoT: Performance evaluation, attacks, optimizations, and protocol integration. *Iraqi Journal of Information and Communication Technology*. 2025. Vol. 8, no. 3. P. 13–35. URL: https://doi.org/10.31987/ijict.8.3.336

[124] Kaur J., Cintas Canto A., Mozaffari Kermani M., Azarderakhsh R. A survey on the implementations, attacks, and countermeasures of the NIST lightweight cryptography standard: Ascon. *ACM Computing Surveys*. 2025. Vol. 58, no. 1. P. 1–16. URL: https://doi.org/10.1145/3744640

[125] Pham-Thi H. T., Bui D. H., Tran X. T. Data communication security for FANETs using Ascon lightweight cryptography. *2025 10th IEEE International Conference on Integrated Circuits, Design, and Verification (ICDV)*. 2025. P. 139–144. URL: https://doi.org/10.1109/ICDV66179.2025.11134863

[126] Rozlomii I., Naumenko S., Trembovetskyi R. Method for rotating cryptographic keys based on time tokens for radio-electronic medical modules with limited resources. *Visnyk NTUU KPI Seriia-Radiotekhnika Radioaparatobuduvannia*. 2025. № 102. С. 58–65. URL: https://doi.org/10.64915/RADAP.2025.102.58-65

[127] Ertaul L., Chauhan A. IoT security: implementation of XTEA, Simon/Speck lightweight block ciphers. *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*. 2023. P. 2478–2485. URL: https://doi.org/10.1109/CSCE60160.2023.00399

[128] Widodo B., Fazrie M., Parulian D. Lightweight cryptography for IoT in wireless sensor networks: Evaluating Speck, Simon, and Ascon using NS-3. *Electronic Journal of Education, Social Economics and Technology*. 2025. Vol. 6, no. 2. P. 1161. URL: https://doi.org/10.33122/ejeset.v6i2.1161

[129] Розломій І. О., Науменко С. В., Симонюк В. В., Пташенчук В. О., Зажома В. В. Полегшена криптографія для безпеки параметрів вібрації в постобробці 3D-друкованих деталей. *Інформаційні технології та суспільство*. 2025. № 2(17). С. 175–182. URL: https://doi.org/10.32689/maup.it.2025.2.25

[130] Mishra Z., Acharya B. Efficient hardware implementation of TEA, XTEA and XXTEA lightweight ciphers for low resource IoT applications. *International Journal of High Performance Systems Architecture*. 2021. Vol. 10, no. 2. P. 80–88. URL: https://doi.org/10.1504/IJHPSA.2021.119150

[131] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The comprehensive IoT security strategy using hardware and software encryption methods. *Materials of the XII International Scientific Conference «Information-Management Systems and Technologies»*, Odesa, 23–25 September 2024. 2024. P. 63–65.

[132] Розломій І. О., Науменко С. В. Моделювання взаємовпливу інформаційної безпеки та обчислювальних витрат у вбудованих пристроях. *Computer-Integrated Technologies: Education, Science, Production*. 2024. № 57. С. 139–145. URL: https://doi.org/10.36910/6775-2524-0560-2024-57-16

[133] Wang R., Yan Y. A survey of secure boot schemes for embedded devices. *2022 24th International Conference on Advanced Communication Technology (ICACT)*. 2022. P. 224–227. URL: https://doi.org/10.23919/ICACT53585.2022.9728840

[134] Annapareddy S. R. Secure boot and firmware authentication mechanisms in embedded devices. *Journal of Engineering and Applied Sciences Technology*. 2023. Vol. 5, no. 4. P. 2–3. URL: https://doi.org/10.47363/JEAST/2023(5)E165

[135] Grübl T., von der Assen J., Knecht M., Stiller B. PACCOR4ESP: Embedded device security attestation using platform attribute certificates. *arXiv*. 2024. URL: https://doi.org/10.48550/arXiv.2407.14286

[136] Schneider K., Auer L., Wagner A. Fault attacks on ECC signature verification. *IACR Transactions on Cryptographic Hardware and Embedded Systems*. 2025. Vol. 2025, no. 4. P. 1010–1052. URL: https://doi.org/10.46586/tches.v2025.i4.1010-1052

[137] Sorensen J. R., Rashid S. A., ElHussini H., Dan A. M. Evaluating differential firmware updates for embedded IoT device fleets. *2025 IEEE 21st International Conference on Factory Communication Systems (WFCS)*. 2025. P. 1–8. URL: https://doi.org/10.1109/WFCS63373.2025.11077647

[138] Çekiş İ. K., Toros A., Apaydın N., Ozcelık İ. Performance comparison of ECC libraries for IoT devices. *Eskişehir Technical University Journal of Science and Technology A – Applied Sciences and Engineering*. 2024. Vol. 25, no. 2. P. 278–288. URL: https://doi.org/10.18038/estubtda.1427488

[139] Harrish A., David S. A lightweight security algorithm for authenticating mobile transactions using wearable devices and ECC based signcryption. *2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS)*. 2024. Vol. 1. P. 1550–1557. URL: https://doi.org/10.1109/ICACCS60874.2024.10717138

[140] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. Modern encryption methods in IoT: hardware solutions and cryptographic libraries for data. *Розвитки інформаційно-керуючих систем та технологій: монографія* / за наук. ред. В. Вичужаніна. Львів–Торунь: Liha-Pres, 2024. P. 28–44. URL: https://doi.org/10.36059/978-966-397-422-4

[141] Hughes L. E. X.509 digital certificate. *Pro Active Directory Certificate Services: Creating and Managing Digital Certificates for Use in Microsoft Networks*. Berkeley: Apress, 2022. P. 45–73. URL: https://doi.org/10.1007/978-1-4842-7486-6_5

[142] Kwon H., Kim D., Kim H., Hahn C., Hur J. Certificate revocation in the TLS ecosystem: A survey. *ACM Computing Surveys*. 2026. Vol. 58, no. 7. P. 1–36. URL: https://doi.org/10.1145/3785653

[143] Nedergaard K., Singh B., Andersen B. Evaluating CoAP, OSCORE, DTLS and HTTPS for secure device communication. *International Conference on Internet of Everything*. Cham: Springer Nature Switzerland, 2022. P. 117–132. URL: https://doi.org/10.1007/978-3-031-25222-8_10

[144] Rozlomii I., Yarmilko A., Naumenko S. Towards distributed anomaly detection in smart networks: a power-efficient node-level approach. *Методи та засоби обчислювального інтелекту в управлінні смарт-системами:* колективна монографія / за заг. ред. В. М. Теслюка. Львів, 2025. С. 109–118.

[145] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovsky P. Hardware encryptors and cryptographic libraries for optimizing security in IoT. *Proceedings of the 12th International Conference Information Control Systems & Technologies (ICST 2024)*. 2024. Vol. 3790. P. 99–109. URL: https://ceur-ws.org/Vol-3790/paper09.pdf

[146] Hercog D., Lerher T., Truntič M., Težak O. Design and implementation of ESP32-based IoT devices. *Sensors*. 2023. Vol. 23, no. 15. P. 6739. URL: https://doi.org/10.3390/s23156739

[147] Kareem H., Dunaev D. The working principles of ESP32 and analytical comparison of using low-cost microcontroller modules in embedded systems design. *2021 4th International Conference on Circuits, Systems and Simulation (ICCSS)*. 2021. P. 130–135. URL: https://doi.org/10.1109/ICCSS51193.2021.9464217

[148] Rathor N., Sinha S. Mender-FPGA: An open source framework for FPGA remote update for ML applications. *2024 IEEE International Symposium on Smart Electronic Systems (iSES)*. 2024. P. 30–35. URL: https://doi.org/10.1109/iSES63344.2024.00017

[149] Srivastava A. K., Kirana C. S., Lilaramani D., Rakshitha R., Sree K. An open-source SWUpdate and Hawkbit framework for OTA updates of RISC-V based

resource constrained devices. *2021 2nd International Conference on Communication, Computing and Industry 4.0 (C2I4)*. 2021. P. 1–6. URL: https://doi.org/10.1109/C2I454156.2021.9689433

[150] Seniushin I., Glazyrina N., Atanbayev Y., Bairamov K., Satiyeva Y., Nurman O., Altaibek M. A framework-driven evaluation and survey of MCU fault injection resilience for IoT. *Applied Sciences*. 2025. Vol. 15, no. 22. P. 11991. URL: https://doi.org/10.3390/app152211991

[151] Mallala B., Tajuddin M. F. N., Thanikanti S. B. Design of power quality analyzer using a field programmable gate array. *Recent Advances in Electrical & Electronic Engineering*. 2025. Vol. 18, no. 8. P. 1321–1334. URL: https://doi.org/10.2174/0123520965314734240606073640

[152] Delgado P., Melo C., Carvalho G., Fukuda V., Silva R. Road surface classification using an accelerometer and MiniROCKET on ultra-low-power STM32. *2025 Sixteenth International Conference on Ubiquitous and Future Networks (ICUFN)*. 2025. P. 235–240. URL: https://doi.org/10.1109/ICUFN65838.2025.11169954

[153] Rozlomii I., Naumenko S., Myhailovskyi P., Lishchuk R. Methodology for selecting the protection strategy in IoT environments based on the device resource profile. *2025 IEEE 6th KhPI Week on Advanced Technology (KhPIWeek)*. 2025. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61436.2025.11288556

[154] Rozlomii I., Yarmilko A., Naumenko S. Integration of lightweight cryptography and artificial intelligence methods to increase the dependability of precision medicine systems. *Proceedings of the International Workshop on Computational Intelligence (IWSCI 2025), co-located with the IV International Scientific Symposium "Intelligent Solutions" (IntSol 2025), Kyiv–Uzhhorod, May 01–05, 2025.* 2025. P. 201–212. URL: https://ceur-ws.org/Vol-4035/Paper17.pdf

[155] Wu D., Liebeherr J. A low-cost low-power LoRa mesh network for large-scale environmental sensing. *IEEE Internet of Things Journal*. 2023. Vol. 10, no. 19. P. 16700–16714. URL: https://doi.org/10.1109/JIOT.2023.3270237

[156] Tamburello M., Caruso G., Adami D., Giordano S. Experimental comparison between SBC and FPGA for embedded neural network acceleration. *ICC 2023 – IEEE International Conference on Communications*. 2023. P. 6078–6083. URL: https://doi.org/10.1109/ICC45041.2023.10279670

[157] Науменко С. В., Розломій І. О., Михайловський П. В. Забезпечення кібербезпеки в Smart-імплантах: роль полегшеної криптографії. *Інформаційна безпека та комп'ютерні технології*: матеріали VII міжнародної науково-практичної конференції, Кропивницький, 1 листоп. 2023 р. Кропивницький, 2023. С. 17–18.

[158] Safari S., Ansari M., Khdr H., Gohari-Nazari P., Yari-Karin S., Yeganeh-Khaksar A., Henkel J. A survey of fault-tolerance techniques for embedded systems from the perspective of power, energy, and thermal issues. *IEEE Access*. 2022. Vol. 10. P. 12229–12251. https://doi.org/10.1109/ACCESS.2022.3144217

[159] Rozlomii O., Yarmilko A., Naumenko S. The intelligent approaches to organizing secure information exchange in dynamic swarms of unmanned platforms. *Artificial Intelligence*. 2024. Vol. 29, no. 4. P. 151–158. URL: https://doi.org/10.15407/jai2024.04.151

[160] Koza E. Semantic analysis of ISO/IEC 27000 standard series and NIST cybersecurity framework to outline differences and consistencies in the context of operational and strategic information security. *Med. Eng. Themes*. 2022. Vol. 2, no. 3. P. 26–39. URL: https://themedicon.com/pdf/engineeringthemes/MCET-02-021.pdf

[161] Rimoli G. P., Gallo L., Fusco P., Ficco M. Power analysis of post-quantum cryptography NIST algorithms in resource-constrained microcontroller. *International Conference on Advanced Information Networking and Applications*. Cham: Springer Nature Switzerland, 2025. P. 161–169. URL: https://doi.org/10.1007/978-3-031-87784-1_15

[162] Ahmet M. High-performance FPGA implementations of lightweight ASCON-128 and ASCON-128a with enhanced throughput-to-area efficiency. *2024*

17th International Conference on Information Security and Cryptology (ISCTürkiye). 2024. P. 1–7. URL: https://doi.org/10.1109/ISCTrkiye64784.2024.10779273

[163] Sun L., Wang W., Wang M. Addendum to linear cryptanalyses of three AEADs with GIFT-128 as underlying primitives. *IACR Transactions on Symmetric Cryptology*. 2022. Vol. 2022, no. 1. P. 212–219. URL: https://doi.org/10.46586/tosc.v2022.i1.212-219

[164] Kundrata J., Baric A. Architectural and post-silicon evaluation of RISC-V vs. Cortex-M0+ SoCs. *2025 MIPRO 48th ICT and Electronics Convention*. 2025. P. 1773–1778. URL: https://doi.org/10.1109/MIPRO65660.2025.11132003

[165] Elsadek I., Tawfik E. Optimized and reconfigurable hardware design for ASCON AEAD and hash standards. *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2025. P. 1–5. URL: https://doi.org/10.1109/ISCAS56072.2025.11043658

[166] Huang L., Zhang J., Yang L., Ma S., Wang Y., Cheng Y. RVAM16: a low-cost multiple-ISA processor based on RISC-V and ARM Thumb. *Frontiers of Computer Science*. 2025. Vol. 19, no. 1. P. 191103. URL: https://doi.org/10.1007/s11704-023-3239-x

[167] Mustafa D., Alkhasawneh R., Obeidat F., Shatnawi A. S. MIMD programs execution support on SIMD machines: A holistic survey. *IEEE Access*. 2024. Vol. 12. P. 34354–34377. URL: https://doi.org/10.1109/ACCESS.2024.3372990

[168] de Araujo Gewehr C. G., Moraes F. G. Improving the efficiency of cryptography algorithms on resource-constrained embedded systems via RISC-V instruction set extensions. *2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)*. 2023. P. 1–6. URL: https://doi.org/10.1109/SBCCI60457.2023.10261964

[169] Gewehr C., Luza L., Moraes F. G. Hardware acceleration of CRYSTALS-Kyber in low-complexity embedded systems with RISC-V instruction set extensions.

IEEE Access. 2024. Vol. 12. P. 94477–94495. URL: https://doi.org/10.1109/ACCESS.2024.3416812

[170] Bogdanov S. G. Secure elements for embedded devices and their applications. *2025 33rd National Conference with International Participation (TELECOM)*. 2025. P. 1–4. URL: https://doi.org/10.1109/TELECOM66943.2025.11304023

[171] Розломій І. О., Науменко С. В. Конфігурований ARX-примітив для енергоефективного хешування у пристроях IoT. *Інформаційні системи та технології: результати і перспективи (IST 2025):* матеріали II Міжнародної науково-практичної конференції. 2025. С. 269–271.

[172] Розломій І. О., Фауре Е. В., Науменко С. В. Методи аутентифікації у вбудованих системах з обмеженими обчислювальними ресурсами. *Вимірювальна та обчислювальна техніка в технологічних процесах*. 2025. № 81. С. 29–35. URL: https://doi.org/10.31891/2219-9365-2025-81-4

[173] Tandel P., Nasriwala J. Efficient authentication framework with Blake2s and a hash-based signature scheme for Industry 4.0 applications. *International Journal of Intelligent Engineering Informatics*. 2024. Vol. 12, no. 4. P. 410–432. URL: https://doi.org/10.1504/IJIEI.2024.142416

[174] Baird I., Wadhaj I., Ghaleb B., Thomson C., Russell G. Evaluating the energy costs of SHA-256 and SHA-3 (KangarooTwelve) in resource-constrained IoT devices. *IoT*. 2025. Vol. 6, no. 3. P. 40. URL: https://doi.org/10.3390/iot6030040

[175] Розломій І. О., Науменко С. В. Модель адаптивної побудови довірчих IoT-мереж із динамічною ротацією вузлів. *Програмне та апаратне забезпечення в інформаційних технологіях:* матеріали Міжнародної науково-практичної конференції молодих вчених та студентів, Луцьк, 6 трав. 2025 р. / відп. ред. Т. В. Терлецький. Луцьк: ЛНТУ, 2025. Вип. 1. С. 136–139.

[176] Yarmilko A., Rozlomii I., Naumenko S. Robust communication clusters: Secure information exchange and redundant hashing for third-party inclusions localization. *КЗЯТПС-2023*. Чернігів, 2023. С. 224–225.

[177] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Модель безпеки взаємопов’язаних обчислювальних пристроїв на основі полегшеної схеми шифрування для IoT. *Computer-Integrated Technologies: Education, Science, Production*. 2024. № 55. С. 191–198. URL: https://doi.org/10.36910/6775-2524-0560-2024-55-24

[178] Розломій І. О., Науменко С. В. Метод розподілу криптографічного навантаження між мікроконтролерами в edge-архітектурах на основі енергетичної моделі. *Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем:* матеріали II Міжнародної науково-практичної конференції. Луцьк: ЛНТУ, 2025. С. 112–115.

ДОДАТКИ

Додаток А. Лістинги розрахунково-експериментальних моделей

А.1. Лістинг програми перевірки хешу та цифрового підпису в bootloader

```
#include <stdint.h>
#include <stddef.h>
#include <stdbool.h>
#include <string.h>
/* Зовнішні криптографічні функції */
extern void sha256(const uint8_t *data, size_t len, uint8_t
hash[32]);
extern bool ecdsa_verify(const uint8_t hash[32],
                         const uint8_t signature[64],
                         const uint8_t public_key[64]);
/* Адреси пам'яті для прикладу */
#define APP_START_ADDRESS      0x08008000U
#define APP_MAX_SIZE           (128U * 1024U)
#define METADATA_SIZE            96U    /* 32 байти хеш + 64 байти
підпис */
/* Публічний ключ виробника / розробника */
static const uint8_t public_key[64] = {
    /* Тут має бути реальний відкритий ключ ECDSA */
    0x12,0x34,0x56,0x78,0x9A,0xBC,0xDE,0xF0,
    0x11,0x22,0x33,0x44,0x55,0x66,0x77,0x88,
    0x10,0x20,0x30,0x40,0x50,0x60,0x70,0x80,
    0x90,0xA0,0xB0,0xC0,0xD0,0xE0,0xF0,0x01,
    0x21,0x43,0x65,0x87,0xA9,0xCB,0xED,0x0F,
    0x13,0x24,0x35,0x46,0x57,0x68,0x79,0x8A,
    0x9B,0xAC,0xBD,0xCE,0xDF,0xE1,0xF2,0x03,
    0x14,0x25,0x36,0x47,0x58,0x69,0x7A,0x8B
```

```
};
/* Метадані прошивки */
typedef struct
{
    uint32_t firmware_size;
    uint32_t firmware_version;
    uint8_t firmware_hash[32];
    uint8_t firmware_signature[64];
} firmware_metadata_t;
/* Допоміжна функція читання метаданих */
static const firmware_metadata_t *get_firmware_metadata(void)
{
    return (const firmware_metadata_t *)(APP_START_ADDRESS +
APP_MAX_SIZE - sizeof(firmware_metadata_t));
}
/* Перевірка коректності розміру прошивки */
static bool is_firmware_size_valid(uint32_t size)
{
    if (size == 0U)
    {
        return false;
    }
    if (size > (APP_MAX_SIZE - sizeof(firmware_metadata_t)))
    {
        return false;
    }
    return true;
}
/* За потреби: захист від downgrade-атаки */
static bool is_version_allowed(uint32_t new_version, uint32_t
min_allowed_version)
{
    return (new_version >= min_allowed_version);
```

```
}
/* Основна функція перевірки прошивки */
bool verify_firmware(void)
{
    const firmware_metadata_t *metadata = get_firmware_metadata();
    const uint8_t *firmware_ptr = (const uint8_t
*)APP_START_ADDRESS;
    uint8_t computed_hash[32];
    /* Перевірка розміру */
    if (!is_firmware_size_valid(metadata->firmware_size))
    {
        return false;
    }
    /* Обчислення SHA-256 для області прошивки */
    sha256(firmware_ptr, metadata->firmware_size, computed_hash);
    /* Перевірка цілісності */
    if (memcmp(computed_hash, metadata->firmware_hash,
sizeof(computed_hash)) != 0)
    {
        return false;
    }
    /* Перевірка автентичності за цифровим підписом */
    if (!ecdsa_verify(metadata->firmware_hash,
                      metadata->firmware_signature,
                      public_key))
    {
        return false;
    }
    return true;
}
/* Передача керування прикладній прошивці */
static void jump_to_application(void)
{
```

```
    uint32_t app_stack = *(volatile uint32_t *)APP_START_ADDRESS;
    uint32_t app_reset = *(volatile uint32_t *)(APP_START_ADDRESS +
4U);
    void (*app_entry)(void) = (void (*)(void))app_reset;
    /* Встановлення основного покажчика стеку */
    __asm volatile ("msr msp, %0" : : "r" (app_stack) : );
    app_entry();
}
/* Головна функція bootloader */
int main(void)
{
    if (verify_firmware())
    {
        jump_to_application();
    }
    else
    {
        /* У разі помилки можна:
           - перейти в безпечний режим;
           - чекати оновлення;
           - сигналізувати про відмову. */
        while (1)
        {
            /* Error state */
        }
    }
    return 0;
}
```

А.2. Лістинг програми генерації криптографічного ключа з використанням апаратного генератора випадкових чисел у вбудованому пристрої

```
#include <stdint.h>
#include <stddef.h>
#include <stdbool.h>
    /* Зовнішні функції платформи STM32 HAL */
extern int HAL_RNG_GenerateRandomNumber(void *hrng, uint32_t
*random_value);
extern void Error_Handler(void);
/* Дескриптор RNG */
typedef struct
{
    void *Instance;
} RNG_HandleTypeDef;
    /* Глобальний дескриптор RNG */
RNG_HandleTypeDef hrng;
    /* Розмір ключа: 128 біт = 16 байт */
#define KEY_SIZE_BYTES 16U
/* Буфер для збереження згенерованого ключа */
static uint8_t session_key[KEY_SIZE_BYTES];
/* Ініціалізація апаратного генератора випадкових чисел */
bool rng_init(void)
{
    /* У проєкті тут виконується:
       - увімкнення тактування RNG;
       - налаштування периферії;
       - виклик HAL_RNG_Init(&hrng); */
    hrng.Instance = (void *)0x50060800U; /* умовна адреса RNG */
    return true;
}
static bool rng_get_word(uint32_t *value)
{
```

```
    if (value == NULL)
    {
        return false;
    }
    if (HAL_RNG_GenerateRandomNumber(&hrng, value) != 0)
    {
        return false;
    }
    return true;
}
bool generate_session_key(uint8_t *key_buffer, size_t key_size)
{
    uint32_t random_word;
    size_t i;
    size_t j;
    if ((key_buffer == NULL) || (key_size == 0U))
    {
        return false;
    }
    for (i = 0; i < key_size; i += 4U)
    {
        if (!rng_get_word(&random_word))
        {
            return false;
        }
        for (j = 0; (j < 4U) && ((i + j) < key_size); j++)
        {
            key_buffer[i + j] = (uint8_t)((random_word >> (8U * j))
& 0xFFU);
        }
    }
    return true;
}
```

Додаток Б. Список публікацій здобувача за темою дисертації та відомості про апробацію результатів дисертації

Наукові праці, в яких опубліковані основні наукові результати дисертації

[1] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. IoT Smart Implants: Information Security and the Implementation of Lightweight Cryptography. *Proceedings of the 6th International Conference on Informatics & Data-Driven Medicine (IDDM'2023).* 2023. P. 145–146. URL: https://ceur-ws.org/Vol-3609/paper12.pdf **(Scopus)**

[2] Rozlomii I., Yarmilko A., Naumenko S. Data security of IoT devices with limited resources: challenges and potential solutions. *Proceedings of the 4th Edge Computing Workshop (DOORS 2024).* 2024. Vol. 3666. P. 85–96. URL: https://ceur-ws.org/Vol-3666/paper13.pdf **(Scopus)**

[3] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The role of encryption in information protection for cloud computing. *IEEE 4th International Conference on Smart Information Systems and Technologies (SIST).* 2024. P. 70–75. URL: https://doi.org/10.1109/SIST61555.2024.10629501 **(Scopus)**

[4] Yarmilko A., Rozlomii I., Naumenko S. Dependability of embedded systems in the Industrial Internet of Things: Information security and reliability of the communication cluster. *Information Technology for Education, Science, and Technics.* 2024. Vol. 222. P. 235–249. URL: https://doi.org/10.1007/978-3-031-71804-5_16 **(Scopus)**

[5] Rozlomii I., Naumenko S., Mykhailovskyi P., Monarkh V. Resource-saving cryptography for microcontrollers in biomedical devices. *IEEE 5th KhPI Week on Advanced Technology (KhPIWeek).* 2024. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61434.2024.10877958 **(Scopus)**

[6] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. Hardware encryptors and cryptographic libraries for optimizing security in IoT. *Proceedings of*

the 12th International Conference Information Control Systems & Technologies (ICST 2024). 2024. Vol. 3790. P. 99–109. URL: https://ceur-ws.org/Vol-3790/paper09.pdf **(Scopus)**

[7] Rozlomii I., Yarmilko A., Naumenko S. Security and efficiency models for cyber-physical systems in medical devices. *IEEE 19th International Conference on Computer Science and Information Technologies (CSIT).* 2024. P. 1–4. URL: https://doi.org/10.1109/CSIT65290.2024.10982678 **(Scopus)**

[8] Rozlomii I., Yarmilko A., Naumenko S. Innovative resource-saving security strategies for IoT devices. *Journal of Edge Computing.* 2025. Vol. 4, no. 1. P. 35–56. URL: https://doi.org/10.55056/jec.748 **(Scopus)**

[9] Rozlomii I., Yarmilko A., Naumenko S. Vulnerability modeling in cybersecurity of intelligent infrastructure networks. *International Scientific-Practical Conference.* 2024. P. 234–248. URL: https://doi.org/10.1007/978-3-031-90735-7_19 **(Scopus)**

[10] Rozlomii I., Yarmilko A., Naumenko S. Resource-efficient solutions for data security at the network level of the Medical Internet of Things. *Proceedings of the 7th International Conference on Informatics & Data-Driven Medicine (IDDM'2024).* 2024. P. 171–182. URL: https://ceur-ws.org/Vol-3892/paper13.pdf **(Scopus)**

[11] Faure E., Rozlomii I., Yarmilko A., Naumenko S. Protection of IoT networks: cryptographic solutions for cybersecurity management. *Proceedings of the Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN 2024).* 2024. P. 24–34. URL: https://ceur-ws.org/Vol-3925/paper03.pdf **(Scopus)**

[12] Rozlomii I., Yarmilko A., Naumenko S. Integration of lightweight cryptography and artificial intelligence methods to increase the dependability of precision medicine systems. *Proceedings of the International Workshop on Computational Intelligence (IWSCI 2025), co-located with the IV International*

Scientific Symposium "Intelligent Solutions" (IntSol 2025), Kyiv–Uzhhorod, May 01–05, 2025. 2025. P. 201–212. URL: https://ceur-ws.org/Vol-4035/Paper17.pdf **(Scopus)**

[13] Faure E., Rozlomii I., Naumenko S. Cryptographic load sharing method in critical infrastructure sensor networks. *Proceedings of the Fourth International Conference on Cyber Hygiene & Conflict Management in Global Information Networks (CH&CMiGIN-25).* 2025. P. 24–34. URL: https://ceur-ws.org/Vol-4024/paper01.pdf **(Scopus)**

[14] Rozlomii I., Naumenko S., Mykhailovskyi P., Lishchuk R. Methodology for selecting the protection strategy in IoT environments based on the device resource profile. *IEEE 6th KhPI Week on Advanced Technology (KhPIWeek).* 2025. P. 1–5. URL: https://doi.org/10.1109/KhPIWeek61436.2025.11288556 **(Scopus)**

[15] Danchenko O., Rozlomii I., Yarmilko A., Naumenko S. A lightweight Secure Boot mechanism for protecting the firmware of IoT devices. *Proceedings of the 13-th International Conference Information Control Systems & Technologies (ICST 2025), Odesa, Ukraine, September 24–26, 2025. CEUR Workshop Proceedings,* Vol. 4048. 2025. P. 240-250. URL: https://ceur-ws.org/Vol-4048/paper19.pdf **(Scopus)**

[16] Rozlomii I., Faure E., Yarmilko A., Naumenko S. The method for verifying firmware integrity in IoT devices for secure boot using lightweight hash functions. *Proceedings of the Cyber Security and Data Protection (CSDP 2025), Lviv, Ukraine, July 31, 2025. CEUR Workshop Proceedings,* Vol. 4042. 2025. P. 105-116. URL: https://ceur-ws.org/Vol-4042/paper8.pdf **(Scopus)**

[17] Rozlomii I., Naumenko S., Trembovetskyi R. Method for rotating cryptographic keys based on time tokens for radio-electronic medical modules with limited resources. *Visnyk NTUU KPI Seriia-Radiotekhnika Radioaparatobuduvannia.* 2025. No. 102. P. 58–65. URL: https://doi.org/10.64915/RADAP.2025.102.58-65 **(Scopus)**

[18] Zabolotnii S., Rozlomii I., Yarmilko A., Naumenko S. Reconfigured CoARX architecture for implementing ARX hashing in microcontrollers of IoT

systems with limited resources. *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska.* 2025. Vol. 15, no. 4. P. 164–169. URL: https://doi.org/10.35784/iapgos.7782 **(Scopus)**

[19] Faure E., Rozlomii I., Naumenko S. Hybrid digital twin-driven anomaly detection in IoT telemetry using LSTM autoencoder. *Proceedings of the 2nd International Workshop on Data Analytics (WDA 2026), Kyiv, Ukraine, January 26, 2026.* CEUR Workshop Proceedings, Vol. 4155. 2025. P. 76–89. URL: https://ceur-ws.org/Vol-4155/paper06.pdf **(Scopus)**

[20] Розломій І. О., Косенюк Г. В., Науменко С. В., Михайловський П. В. Моделювання системи датчиків на базі мікроконтролера в ігровій симуляції «Смарт-будинок» з використанням шифрування. *Computer-Integrated Technologies: Education, Science, Production.* 2023. Вип. 53. С. 292–299. URL: https://doi.org/10.36910/6775-2524-0560-2023-53-43

[21] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Адаптивна криптографія для енергоефективного захисту пристроїв IoT. *Проблеми моделювання та автоматизації проектування.* 2024. № 1(19). С. 77–83. URL: https://doi.org/10.31474/2074-7888-2024-1-19-77-83

[22] Розломій І. О., Симонюк В. П., Науменко С. В., Михайловський П. В. Модель безпеки взаємопов’язаних обчислювальних пристроїв на основі полегшеної схеми шифрування для IoT. *Computer-Integrated Technologies: Education, Science, Production.* 2024. Вип. 55. С. 191–198. URL: https://doi.org/10.36910/6775-2524-0560-2024-55-24

[23] Rozlomii O., Yarmilko A., Naumenko S. The intelligent approaches to organizing secure information exchange in dynamic swarms of unmanned platforms. *Artificial Intelligence.* 2024. Vol. 29, no. 4. P. 151–158. URL: https://doi.org/10.15407/jai2024.04.151

[24] Розломій І. О., Науменко С. В. Моделювання взаємовпливу інформаційної безпеки та обчислювальних витрат у вбудованих пристроях.

Computer-Integrated Technologies: Education, Science, Production. 2024. Вип. 57. С. 139–145. URL: https://doi.org/10.36910/6775-2524-0560-2024-57-16

[25] Розломій І. О., Фауре Е. В., Науменко С. В. Методи аутентифікації у вбудованих системах з обмеженими обчислювальними ресурсами. *Вимірювальна та обчислювальна техніка в технологічних процесах.* 2025. Вип. 81. С. 29–35. URL: https://doi.org/10.31891/2219-9365-2025-81-4

[26] Розломій І. О., Науменко С. В. Архітектура та функціональні особливості захищених систем керування базами даних нового покоління з підтримкою serverless та edge-обчислень. *Systems and Technologies.* 2025. №1 (69), С. 7–15. URL: https://doi.org/10.32782/2521-6643-2025-1-69.16

[27] Розломій І. О., Науменко С. В., Симонюк В. В., Пташенчук В. О., Зажома В. В. Полегшена криптографія для безпеки параметрів вібрації в постобробці 3D-друкованих деталей. *Інформаційні технології та суспільство.* 2025. № 2(17). С. 175–182. URL: https://doi.org/10.32689/maup.it.2025.2.25

[28] Rozlomii I., Koseniuk G., Naumenko S. Mechanisms for cryptographic code authentication control in sensor nodes with limited computing resources. *Computer-Integrated Technologies: Education, Science, Production.* 2025. No. 61. P. 193–198. URL: https://doi.org/10.36910/6775-2524-0560-2025-61-27

[29] Розломій І., Науменко С., Ковтюх В. Модель захищеного зберігання даних у розподілених базах даних на основі атрибутного шифрування для критичних інформаційно-комунікаційних систем. *Measuring and Computing Devices in Technological Processes.* 2026. № 1. С. 215–220. URL: https://doi.org/10.31891/2219-9365-2026-85-27

[30] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. Modern encryption methods in IoT: hardware solutions and cryptographic libraries for data. *Розвитки інформаційно-керуючих систем та технологій: монографія / за ред. В. Вичужаніна. Львів-Торунь: Liha-Pres,* 2024. P. 28–44. URL: https://doi.org/10.36059/978-966-397-422-4

[31] Розломій І. О., Ярмілко А. В., Науменко С. В. Ресурсоощадний підхід до побудови secure boot у вбудованих системах інтернету речей. *Системи контролю інформації та інтелектуальні технології. Досягнення та застосування: монографія / за ред. В. Вичужаніна. Львів-Торунь: Liha-Pres,* 2025. С. 102–122. URL: https://doi.org/10.36059/978-966-397-538-2-6

[32] Rozlomii I., Yarmilko A., Naumenko S. Towards distributed anomaly detection in smart networks: a power-efficient node-level approach. *Методи та засоби обчислювального інтелекту в управлінні смарт-системами: колективна монографія / за ред. В. М. Теслюка.* Львів, 2025. С. 109–118.

Наукові праці, які засвідчують апробацію матеріалів дисертації

[1] Yarmilko A., Rozlomii I., Naumenko S. Robust communication clusters: Secure information exchange and redundant hashing for third-party inclusions localization. *КЗЯТПС-2023:* тези доп. Чернігів, 2023. Р. 224–225.

[2] Науменко С. В., Розломій І. О. Information protection strategies in Industry 4.0: Encryption and cybersecurity for industrial systems. *Theoretical and Experimental Research in Materials Science and Mechanical Engineering:* матеріали IX Міжнар. наук.-практ. конф., Луцьк, 2023. Луцьк: Вежа-Друк, 2023. С. 191–193.

[3] Науменко С. В., Розломій І. О., Михайловський П. В. Забезпечення кібербезпеки в Smart-імплантах: роль полегшеної криптографії. *Інформаційна безпека та комп'ютерні технології:* матеріали VII Міжнар. наук.-практ. конф., м. Кропивницький, 1 листоп. 2023 р. Кропивницький, 2023. С. 17–18.

[4] Rozlomii I. O., Yarmilko A., Naumenko S. Optimized hash functions for integrity control and data recovery in embedded systems. *Information-Management Systems and Technologies:* матеріали XI Міжнар. наук. конф. 2023. С. 31–34.

[5] Rozlomii I., Yarmilko A., Naumenko S., Mykhailovskyi P. The comprehensive IoT security strategy using hardware and software encryption methods.

Information-Management Systems and Technologies: матеріали XII Міжнар. наук. конф., Одеса, 23 – 25 верес. 2024 р. Одеса, 2024. С. 63–65.

[6] Розломій І. О., Ярмілко А. В., Науменко С. В. Інтелектуальні підходи до організації інформаційного обміну в динамічних зграях безпілотних платформ. *Штучний інтелект та інтелектуальні системи (AIIS'2024):* матеріали XXIV Міжнар. наук.-техн. конф. 2024. С. 144–149.

[7] Науменко С. В., Михайловський П. В., Стабецька Т. А. Сучасні технології захисту даних у вбудованих пристроях з обмеженими ресурсами. *Free and Open Source Software:* матеріали Міжнар. наук.-практ. конф., Харків, 13 – 14 лют. 2025 р. Харків, 2025. С. 63–64.

[8] Розломій І. О., Науменко С. В. Конфігурований ARX-примітив для енергоефективного хешування у пристроях IoT. *Інформаційні системи та технології: результати і перспективи (IST 2025):* матеріали II Міжнар. наук.-практ. конф. 2025. С. 269–271.

[9] Чікін Д. М., Науменко С. В., Розломій І. О. Захист персональних даних в IoT-пристроях із застосуванням штучного інтелекту. *Інформаційна безпека та комп'ютерні технології:* тези доп. VIII Міжнар. наук.-практ. конф., м. Кропивницький, 24 – 25 квіт. 2025 р. Кропивницький: ЦНТУ, 2025. С. 12–13.

[10] Розломій І. О., Науменко С. В., Михайловський П. В. Формування сеансових ключів у сенсорних пристроях з обмеженим обсягом пам'яті на основі часових токенів. *Комплексне забезпечення якості технологічних процесів та систем (КЗЯТПС – 2025):* матеріали XV Міжнар. наук.-практ. конф., Чернігів, 22 – 23 трав. 2025 р. Чернігів: НУ «Чернігівська політехніка», 2025. Вип. 2. С. 248–249.

[11] Розломій І. О., Науменко С. В. Метод розподілу криптографічного навантаження між мікроконтролерами в edge-архітектурах на основі енергетичної моделі. *Проблеми комп'ютерних наук, програмного моделювання*

та безпеки цифрових систем: матеріали II Міжнар. наук.-практ. конф., Луцьк, 2025. Луцьк: ЛНТУ, 2025. С. 112–115.

[12] Розломій І. О., Науменко С. В. Модель адаптивної побудови довірчих IoT-мереж із динамічною ротацією вузлів. *Програмне та апаратне забезпечення в інформаційних технологіях:* матеріали Міжнар. наук.-практ. конф. молодих вчених та студентів, Луцьк, 6 трав. 2025 р. / відп. ред. Т. В. Терлецький. Луцьк: ЛНТУ, 2025. Вип. 1. С. 136–139.