УДК 004.056.55
DOI https://doi.org/10.32782/2663-5941/2026.3.1/26

Онай М.В.
https://orcid.org/0000-0002-4938-8355
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Згуровський Я.Ю.
https://orcid.org/0009-0006-6087-1643
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

ПРОЄКТУВАННЯ АРХІТЕКТУРИ ТА ФОРМУЛЮВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ ОЦІНЮВАННЯ ПОСТКВАНТОВИХ КРИПТОГРАФІЧНИХ АЛГОРИТМІВ

Стаття присвячена розв'язанню актуальної науково-практичної задачі забезпечення безпечної міграції сучасних інформаційно-комунікаційних систем на постквантову криптографію (PQC). Стрімкий розвиток квантових обчислень та ймовірність швидкої появи криптографічно значущого квантового комп'ютера створюють критичну загрозу для класичних асиметричних криптосистем (RSA, ECC), що вимагає переходу до нових стандартизованих алгоритмів. З'ясовано, що практичне впровадження алгоритмів постквантової криптографії є надзвичайно складною інженерною проблемою. На відміну від класичної криптографії, постквантові алгоритми вимагають значно більших розмірів ключів та підписів, що призводить до фрагментації мережевих пакетів, а також характеризуються підвищеним споживанням процесорного часу та пам'яті.

Аналіз останніх досліджень показав, що існуючі інструменти профілювання є фрагментованими, переважно тестують алгоритми в ізоляції та не здатні комплексно врахувати вплив PQC на мережеві протоколи або виявити апаратні вразливості до атак побічними каналами при виконанні на цільових мікроархітектурах.

Метою роботи є формування комплексу вимог та проєктування цілісної концептуальної архітектури програмної системи для оцінювання застосування постквантових криптографічних алгоритмів.

У межах дослідження розроблено формалізовану специфікацію вимог, у якій деталізовано функціональні, безпекові та інтерфейсні вимоги, серед яких ключовими є підтримка криптографічної гнучкості, генерація гібридних криптосхем для перехідного періоду, проведення математичної валідації та забезпечення точності апаратної телеметрії. Взаємодію акторів із системою змодельовано за допомогою діаграми прецедентів UML. Запропоновано архітектурний концепт програмної системи, побудований на основі методології C4 із застосуванням парадигми Architecture as Code. Архітектуру деталізовано на трьох рівнях: системного контексту, контейнерів та компонентів. На рівні контейнерів систему розділено на модулі криптографічної абстракції, апаратного профілювання, генерації звітів та модуль мережевої емуляції, що дозволяє імітувати зниження продуктивності мережі. На рівні компонентів здійснено розкладення операційного ядра програмної системи.

Наукова новизна одержаних результатів полягає у створенні комплексної архітектурної моделі, яка поєднує ізольоване апаратне профілювання, тестування гібридних криптосхем та мережеву емуляцію в єдиному інструментальному середовищі з чітко визначеними межами відповідальності компонентів.

Практична значущість полягає у тому, що запропонована архітектура вирішує проблему інженерної фрагментованості існуючих утиліт та створює надійне підґрунтя для програмної реалізації масштабованої програмної системи. Це дозволить користувачам здійснювати обґрунтований вибір оптимальних алгоритмів, адаптованих до обмежень різних платформ.

Ключові слова: *постквантова криптографія, програмне забезпечення, архітектура програмного забезпечення, програмна система, еліптична криптографія, модель C4.*

Постановка проблеми. Сучасна архітектура глобальної інформаційної безпеки, яка забезпечує конфіденційність та цілісність цифрових комунікацій, спирається на криптографію з відкритим ключем (Public Key Cryptography, PKC). Фундаментальні механізми безпеки класичної асиметричної криптографії ґрунтуються на обчислювальній складності конкретних математичних задач. Для найпоширеніших сьогодні криптосистем, таких як RSA та алгоритмів на основі еліптичних кривих, криптографічна стійкість забезпечується складністю задачі факторизації великих цілих чисел та задачі дискретного логарифмування відповідно [1]. Проте стрімкий технологічний розвиток у галузі квантових обчислень створює загрозу, яка полягає в тому, що криптографічно значущий квантовий комп'ютер (CRQC) здатен розв'язувати ці задачі за поліноміальний час. Це означає повну компрометацію існуючих стандартів [2].

Наукова спільнота готується до події «Y2Q» (Years to Quantum). Аналітичні прогнози вказують на високу ймовірність появи CRQC до 2031 року. Більше того, глобальна кібербезпека вже сьогодні стикається зі стратегією «Harvest Now, Decrypt Later» (збережи зараз, розшифруй пізніше), коли зловмисники накопичують зашифровані масиви даних в очікуванні квантових потужностей [3]. У відповідь на це Національний інститут стандартів і технологій США (NIST) ініціював процес стандартизації постквантової криптографії (PQC), який у серпні 2024 року завершився публікацією перших стандартів: FIPS 203 (ML-KEM), FIPS 204 (ML-DSA) та FIPS 205 (SLH-DSA) [4]. Паралельно впроваджуються регуляторні директиви, такі як алгоритмічний набір CNSA 2.0 від Агентства національної безпеки США (NSA), що вимагає переходу на нові стандарти для підпису вже з 2025 року та повної відмови від класичної криптографії до 2030-2035 років [5]. Подібні дорожні карти впроваджуються і в ЄС під керівництвом ENISA [6].

Проте практичне впровадження алгоритмів PQC є надзвичайно складною інженерною проблемою. На відміну від класичної криптографії, алгоритми PQC вимагають значно більших розмірів відкритих і закритих ключів, цифрових підписів (що призводить до фрагментації мережевих пакетів), а також характеризуються підвищеним споживанням процесорного часу та оперативної пам'яті. Наприклад, алгоритми на основі ґраток забезпечують високу швидкість, але створюють значне навантаження на стек пам'яті, що є критичним для IoT-пристроїв. Відтак, виникає важливе практичне завдання – щоб уникнути зниження продуктивності мережевих протоколів (таких як TLS 1.3, MQTT, CoAP) і забезпечити стабільну роботу інфраструктури, необхідне комплексне, багатокритеріальне оцінювання PQC-алгоритмів безпосередньо в умовах цільових апаратних та мережевих платформ.

Аналіз останніх досліджень і публікацій. Дослідження ефективності постквантових криптографічних алгоритмів наразі є одним із найактивніших напрямів у галузі кібербезпеки. Аналіз сучасного наукового та інженерного доробку дозволяє виділити кілька ключових підходів до оцінювання PQC, кожен з яких має свої переваги та суттєві обмеження.

Ізольоване математичне та апаратне профілювання. Історично оцінювання криптографічних примітивів здійснювалося за допомогою монолітних наборів для оцінювання, таких як проєкт SUPERCOP, який забезпечує порівняльний аналіз швидкодії алгоритмів. Розвитком цього підходу став проєкт PQClean, що надає еталонні реалізації алгоритмів мовою C, оптимізовані для архітектур Intel та ARMv8, дозволяючи тестувати їх в ізоляції від системного середовища [7]. Аналогічну роль для мікроконтролерів відіграє проєкт pqm^4, призначений для платформ ARM Cortex-M4. Однак дослідження показують, що ізольоване тестування може приховувати апаратні вразливості. Наприклад, автори аналізу алгоритму Falcon на платах STM32 з процесором Cortex-M7 виявили, що використання вбудованого модуля операцій з плаваючою комою (FPU) хоч і прискорює виконання у 6.2-8.3 рази, але призводить до порушення принципу виконання за постійний час (constant-time irregularities), відкриваючи вектор для атак побічними каналами (SCA) [8].

Бібліотеки інтеграції та гібридні фреймворки. Найпоширенішим інструментом для інтеграції PQC у прикладні рішення є проєкт liboqs (Open Quantum Safe), який надає уніфікований API та дозволяє інтегрувати алгоритми у відгалуження бібліотеки OpenSSL [9]. На базі бібліотеки liboqs створюються нові інструменти оцінювання, такі як фреймворк PQC-LEO, що автоматизує збір метрик продуктивності та мережевих затримок протоколу TLS 1.3 на архітектурах x86 та ARM [10]. Крім того, дослідники все частіше наголошують на необхідності перехідного етапу у вигляді гібридних криптографічних фреймворків (Hybrid Cryptographic Framework, HCF), які поєднують класичні (наприклад, ECDH) та PQC-алгоритми

(наприклад, ML-KEM) для мінімізації ризиків. Проте самі розробники liboqs зазначають, що їхнє програмне забезпечення призначене переважно для прототипування і не містить повноцінного захисту від вразливостей пам'яті для використання у реальних середовищах.

Мережева емуляція та вплив на протоколи. Окремим напрямом є дослідження PQC у розподілених мережах, зокрема в IoT. Сучасні дослідження вказують на критичну прогалину: більшість тестів фокусується лише на протоколі TLS, ігноруючи специфіку IoT. Для вирішення цього завдання було запропоновано фреймворки за принципом Software-in-the-Loop (SITL), такі як PQC-IoTNet, що дозволяють тестувати PQC-алгоритми поверх легковагових протоколів MQTT та CoAP. Такі платформи виявили, що в умовах навіть 5% втрати пакетів у бездротових мережах, великі розміри ключів PQC призводять до нелінійних сплесків затримок (latency spikes) через ретрансмісію на рівні TCP [11].

Незважаючи на значний прогрес та наявність інструментів (liboqs, PQClean, PQC-LEO, SITL-симулятори), поточний ландшафт програмного забезпечення для оцінювання PQC залишається фрагментованим. Більшість існуючих рішень вирішують вузькоспеціалізовані задачі, такі як тестування швидкодії, або виключно мережевої взаємодії.

На сьогодні відсутня комплексна, уніфікована архітектура програмної системи, яка б органічно об'єднувала: абстракцію криптографічних примітивів (Crypto-Agility); розширене системне профілювання; мережеву емуляцію. Крім того, в існуючих публікаціях практично відсутній формалізований архітектурний опис таких систем на рівні компонентів, що критично ускладнює їх інтеграцію у сучасні корпоративні екосистеми, середовища CI/CD та процеси міграції великих інформаційних систем на квантово-стійкі стандарти.

Постановка завдання. Метою цього дослідження є формулювання комплексу вимог та проєктування концептуальної архітектури програмної системи для оцінювання застосування постквантових криптографічних алгоритмів з використанням архітектурної методології C4. Досягнення цієї мети передбачає формалізацію функціональних та нефункціональних вимог до програмної системи, що враховують специфіку постквантової криптографії (криптографічна гнучкість, гібридні режими, накладні мережеві витрати) та апаратну гетерогенність, а також розроблення архітектурного концепту програмної системи на основі моделі C4, що забезпечить чітке розмежування відповідальності між модулями криптографічної абстракції, апаратного профілювання та мережевої емуляції, створюючи основу для подальшої програмної реалізації.

Виклад основного матеріалу. *Специфікація вимог до програмної системи оцінювання.* Для забезпечення здатності розроблюваної програмної системи вирішувати комплексні задачі профілювання в гетерогенних середовищах, процес проєктування її архітектури повинен спиратися на формалізовану специфікацію вимог. Систематизований перелік вимог, розподілений за категоріями, наведено у таблиці 1.

Взаємодія зовнішніх акторів із розроблюваною системою описується діаграмою прецедентів UML (рис. 1). Основними акторами виступають «Інженер з кібербезпеки» (особа, яка ініціює спеціалізовані бенчмарки) та «Система CI/CD» (автоматизований конвеєр збірки, що запускає тестування при оновленні коду).

Як видно з діаграми, запуск будь-яких тестів продуктивності обов'язково включає етап попередньої валідації математичної коректності алгоритмів, що відповідає вимозі FR-04. Сценарії мережевої емуляції та гібридного тестування є розширеннями процесу базового налаштування конфігурації.

Архітектурний концепт системи на основі моделі C4. Модель C4 (Context, Containers, Components, Code) – це сучасний підхід до документування архітектури програмного забезпечення, створений розробником Саймоном Брауном [12]. Вона пропонує ієрархічний набір абстракцій, що дозволяє описувати систему на різних рівнях деталізації, подібно до того, як працюють цифрові карти (від глобального масштабу карти світу до окремих будівель). Основна перевага C4 полягає в орієнтації на різні цільові аудиторії: наприклад, рівень контексту зрозумілий бізнес-стейкхолдерам, а рівні компонентів та коду призначені безпосередньо для розробників. Це забезпечує ефективну комунікацію, покращує дизайн та полегшує підтримку системи. У даному дослідженні використано перші три рівні моделі, які є достатніми для проєктування повноцінної концептуальної архітектури програмної системи.

Замість використання статичних діаграм загального призначення архітектура реалізована за принципом Architecture as Code (AaC) з використанням спеціалізованої предметно-орієнтованої мови Structurizr DSL. Це гарантує узгодженість структур на всіх рівнях абстракції.

Таблиця 1

Вимоги до програмної системи

ID	Категорія	Назва вимоги	Детальний опис (специфікація)
FR-01	Функціональна	Універсальна підтримка KEM та DSA	Система повинна реалізовувати стандартизовані API для повного життєвого циклу KEM (KeyGen, Encaps, Decaps) та DSA (KeyGen, Sign, Verify) незалежно від базового математичного типу алгоритму (ґратки, хеші, коди).
FR-02	Функціональна	Криптографічна гнучкість	Забезпечення можливості динамічного перемикання криптографічних алгоритмів або їх провайдерів через конфігураційні файли (наприклад, JSON/YAML формат) без перекомпіляції ядра системи.
FR-03	Функціональна	Підтримка гібридних криптосхем	Система повинна забезпечити можливість конструювати композитні примітиви (наприклад, класичний ECDH в поєднанні з постквантовим ML-KEM) для тестування перехідних сценаріїв згідно з рекомендаціями NIST SP 800-227.
FR-04	Функціональна	Математична валідація	Перед запуском будь-якого бенчмарку система зобов'язана провести тестування алгоритму за векторами з відомими відповідями (Known Answer Tests) для підтвердження його математичної коректності.
FR-05	Функціональна	Мережева емуляція	Вбудований симулятор повинен перехоплювати PQC-трафік (TLS 1.3, MQTT, CoAP) та імітувати зниження продуктивності мережі (до 10% втрати пакетів, затримки) для вимірювання впливу фрагментації PQC-ключів.
PR-01	Продуктивність	Точність апаратної телеметрії	Збір часових метрик повинен здійснюватися шляхом прямого доступу до апаратних лічильників циклів процесора (Hardware Performance Counters), ігноруючи системні таймери ОС, з похибкою не більше ±1%.
PR-02	Продуктивність	Мінімізація впливу спостерігача	Програмні запити профілювальника не повинні споживати більше 2% ресурсів процесора та пам'яті під час виконання тестованого алгоритму (аналіз системи, який не впливає на її роботу під час процесу вимірювання).
PR-03	Продуктивність	Масштабованість тестування	Система повинна підтримувати паралельну генерацію не менше 10 000 одночасних TLS-рукостискань для імітації стресового навантаження у хмарних архітектурах.
SR-01	Безпека	Профілювання постійного часу	Аналізатор повинен виявляти порушення принципу constant-time execution (варіативність часу виконання залежно від секретних даних) для оцінки вразливості до атак побічними каналами (SCA).
IR-01	Інтерфейси	Інтеграція з CI/CD (API/CLI)	Програмна система повинна надавати інтерфейс командного рядка (CLI) для запуску з конвеєрів збірки (Jenkins, GitLab CI) та експортувати результати у машиночитабельному форматі (JSON, CSV).
IR-02	Інтерфейси	Шар апаратних абстракцій	Наявність стандартизованого інтерфейсу для розгортання цільового коду як на серверах (x86_64), так і на ресурсообмежених мікроконтролерах (ARM Cortex-M/A).

Наведені нижче візуалізації архітектури описують декомпозицію системи на трьох етапах:

1. Контекст системи (Context) – цей рівень є найвищою точкою огляду. Він відображає розроблювану систему як єдиний «чорний ящик» і фокусується на її взаємодії із зовнішнім світом (рис. 2). Головна мета цього рівня показати, яку цінність несе система, хто є її основними акторами та від яких зовнішніх сутностей вона залежить. У нашій моделі акторами виступають інженери з безпеки та конвеєри CI/CD, а зовнішньою залежністю є цільова апаратна платформа.

2. Контейнери (Container) – на цьому рівні монолітна система розкривається і розкладається на окремі розгортувані одиниці (контейнери) – такі як веб-сервери, бази даних або окремі виконувані модулі (рис. 3). Цей рівень показує високорівневі технологічні рішення та те, як окремі частини системи розподіляють обов'язки і спілкуються між собою. У запропонованій архітектурі виділено спеціалізовані модулі: інтерфейс керування (CLI), оркестратор, криптографічний шар, мережевий емулятор та профілювальник.

3. Компоненти (Component) – цей рівень заглиблюється у структуру конкретного контейнера, розбиваючи його на логічні структурні блоки (модулі, бібліотеки, контролери) (рис. 4). Він деталізує інкапсуляцію бізнес-логіки. Для програмної системи було здійснено розкладення найважливішого контейнера – Core Benchmarking Engine, розділивши його на логічні підсистеми

Рис. 1. Діаграма прецедентів

Рис. 2. Контекст системи

управління, генерації навантаження та агрегації телеметрії.

Функціональне призначення визначених контейнерів та компонентів відповідає розробленій специфікації:

– API / CLI Interface – реалізує вимогу IR-01, виступаючи єдиною точкою входу.

– Crypto Agility Layer – шар абстракції (FR-01, FR-02), що ізолює ядро від специфіки конкретних бібліотек та реалізує логіку конструювання гібридних криптосхем (FR-03).

– Network Emulation Container (SITL) – забезпечує моделювання втрат пакетів та затримок (FR-05).

– Hardware Profiling Module – забезпечує виконання продуктивних та безпекових вимог (PR-01, PR-02, SR-01). Звертається до HAL (IR-02) цільової платформи для точного вимірювання тактів та споживання пам'яті.

– Core Benchmarking Engine – відокремлює логіку підготовки навантаження (Workload Generator), обов'язкової математичної перевірки алгоритмів (KAT Validator для вимоги FR-04) та збору результатів (Telemetry Aggregator). Це гарантує високу розширюваність системи.

Висновки. Глобальна міграція IT інфраструктури на постквантову криптографію вимагає від інженерів обґрунтованого вибору алгоритмів для кожної конкретної задачі. Відсутність цілісних архітектурних рішень для профілювання PQC ускладнює цей процес.

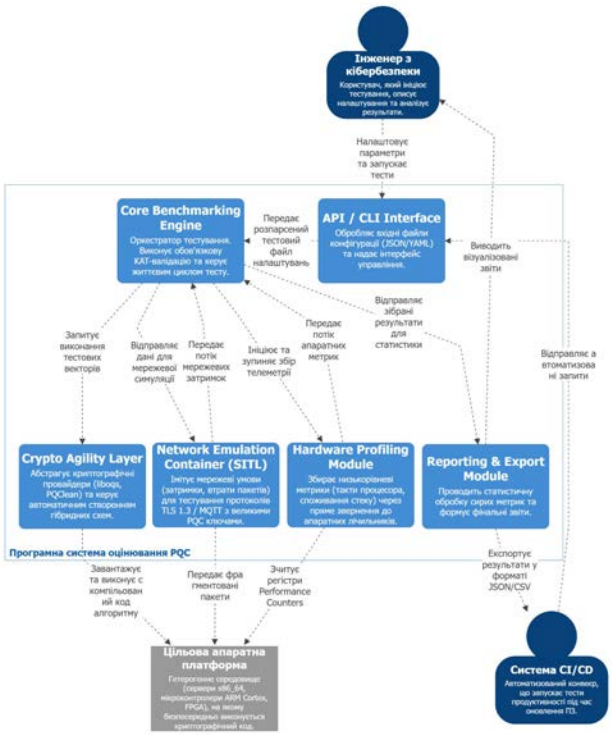

Рис. 3. Рівень контейнерів

У межах цього дослідження формалізовано комплексну специфікацію вимог до програмної системи оцінювання PQC, з деталізацією функціональних характеристик, атрибутів продуктивності, безпеки та системних інтерфейсів. На основі цих вимог розроблено архітектурний концепт програмної системи за допомогою методології C4, яка дозволяє представити складну систему на різних рівнях абстракції: від глобального контексту до структури конкретних компонентів. Запропонована архітектура вирішує проблему фрагментованості існуючого інструментарію шляхом інтеграції модулів криптографічної абстракції (Crypto-Agility), точного апаратного профілювання та мережевої емуляції (SITL) в єдину цілісну систему. Розбиття операційного ядра на рівні компонентів підтвердила життєздатність та гнучкість обраного підходу.

Подальші перспективи дослідження полягають у проєктуванні фізичних структур баз даних для збереження результатів тестування та безпосередній програмній реалізації запропонованої програмної системи для проведення експериментів.

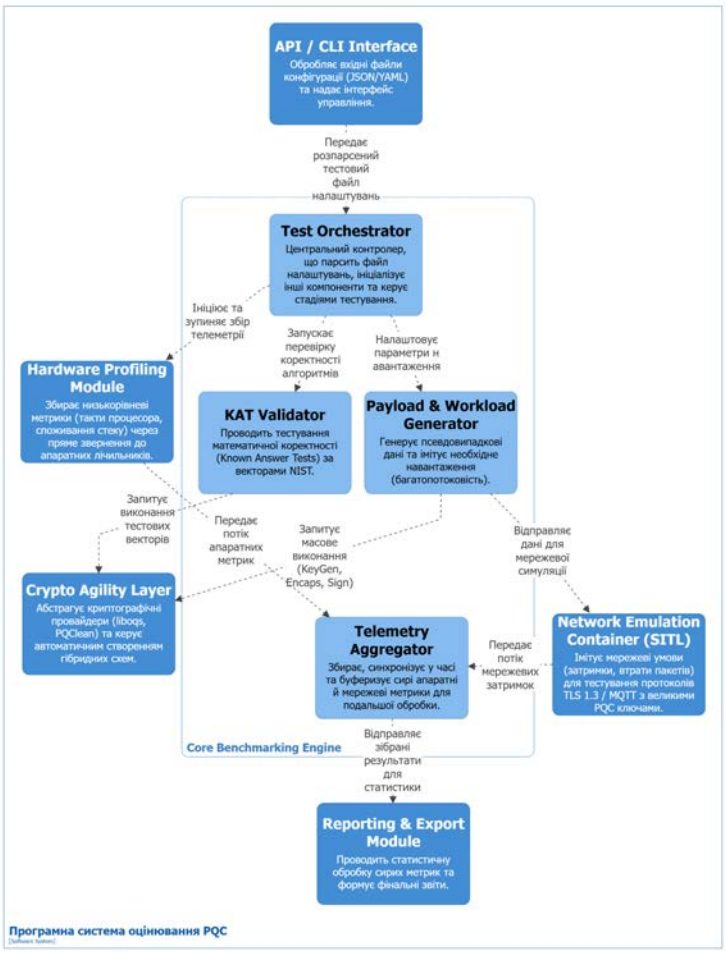

Рис. 4. Рівень компонентів ядра оцінювання

Список літератури:

1. Hasan K. F. et al. A Framework for Migrating to Post-Quantum Cryptography: Security Dependency Analysis and Case Studies. *IEEE Access*. 2024. Vol. 12. P. 23427–23450. DOI: 10.1109/ACCESS.2024.3360412.
2. Ahmed N., Zhang L., Gangopadhyay A. A Survey of Post-Quantum Cryptography Support in Cryptographic Libraries. *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)* (Albuquerque, NM, USA, 2025). 2025. P. 906–917. DOI: 10.1109/QCE65121.2025.00102.
3. Barker W., Polk W., Souppaya M. Getting ready for post-quantum cryptography: Exploring challenges associated with adopting and using post-quantum cryptographic algorithms (NIST CSWP 04282021). National Institute of Standards and Technology. 2021. DOI: 10.6028/NIST.CSWP.04282021.
4. Post-quantum cryptography. National Institute of Standards and Technology. 2025. URL: https://csrc.nist.gov/projects/post-quantum-cryptography (дата звернення: 19.03.2026).
5. Announcing the Commercial National Security Algorithm Suite 2.0. National Security Agency (NSA). 2025. URL: https://media.defense.gov/2025/May/30/2003728741/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS.PDF (дата звернення: 19.03.2026).
6. Beullens W. et al. Post-Quantum Cryptography: Current state and quantum mitigation. European Union Agency for Cybersecurity (ENISA). 2021. URL: https://www.enisa.europa.eu/sites/default/files/publications/ENISA%20Report%20-%20Post-Quantum%20Cryptography%20Current%20state%20and%20quantum%20mitigation-V2.pdf (дата звернення: 19.03.2026).
7. Kannwischer M. J., Schwabe P., Stebila D., Wiggers T. Improving Software Quality in Cryptography Standardization Projects. *2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (Genoa, Italy, 2022). 2022. P. 19–30. DOI: 10.1109/EuroSPW55150.2022.00010.
8. Howe J., Westerbaan B. Benchmarking and Analysing the NIST PQC Lattice-Based Signature Schemes Standards on the ARM Cortex M7. *Progress in Cryptology – AFRICACRYPT 2023*. Cham : Springer, 2023. Vol. 14064. DOI: 10.1007/978-3-031-37679-5_19.
9. liboqs | Open Quantum Safe. Open Quantum Safe a Series of Linux Foundation Projects, LLC. URL: https://openquantumsafe.org/liboqs/ (дата звернення: 19.03.2026).
10. GitHub - PQC-LEO. GitHub, Inc. URL: https://github.com/crt26/PQC-LEO (дата звернення: 19.03.2026).
11. Almutairi M. G., Sheldon F. T. Methodology and Architecture for Benchmarking End-to-End PQC Protocol Resilience in an IoT Context. *IoT*. 2026. Vol. 7, No. 1. Art. 17. DOI: 10.3390/iot7010017.
12. Brown S. C4 model for visualising software architecture. URL: https://c4model.com/ (дата звернення: 19.03.2026).

Onai M.V., Zghurovskyi Ya.Yu. ARCHITECTURE DESIGN AND REQUIREMENTS SPECIFICATION OF A SOFTWARE SYSTEM FOR POST-QUANTUM CRYPTOGRAPHIC ALGORITHM EVALUATION

The article is devoted to solving an urgent scientific and practical problem of ensuring the secure migration of modern information and communication systems to post-quantum cryptography (PQC). The rapid development of quantum computing and the likelihood of the imminent emergence of a cryptographically relevant quantum computer pose a critical threat to classical asymmetric cryptosystems (RSA, ECC), necessitating a transition to new standardized algorithms. It was found that the practical implementation of post-quantum cryptographic algorithms is an extremely complex engineering challenge. Unlike classical cryptography, post-quantum algorithms require significantly larger key and signature sizes, leading to network packet fragmentation, as well as increased CPU and memory consumption.

Analysis of recent research shows that existing profiling tools are fragmented, mostly testing algorithms in isolation, and are unable to comprehensively account for the impact of PQC on network protocols or identify hardware side-channel attack vulnerabilities when executed on target microarchitectures.

The aim of this research is to formulate a comprehensive set of requirements and design a holistic conceptual architecture for a software system to evaluate the application of post-quantum cryptographic algorithms.

Within the research, a formalized requirements specification was developed, detailing functional, security, and interface requirements. Key requirements include support for cryptographic agility, generation of hybrid cryptosystems for the transition period, mathematical validation, and ensuring the accuracy of hardware telemetry. Actor interaction with the system is modeled using a UML Use Case diagram. An architectural concept for the software system is proposed, built on the C4 methodology using the "Architecture as Code" paradigm. The architecture is detailed at three levels: system context, containers, and components. At the container level, the system is decomposed into modules for cryptographic abstraction, hardware profiling, report generation, and network emulation to simulate network degradation. At the component level, the operational core of the software system was decomposed.

The scientific novelty of the results lies in the creation of a comprehensive architectural model that integrates isolated hardware profiling, hybrid cryptosystem testing, and network emulation within a single instrumental environment with clearly defined component boundaries. The practical significance is that the proposed architecture addresses the engineering fragmentation of existing utilities and creates a solid foundation for the implementation of a scalable software system. This will enable users to make informed choices of optimal algorithms adapted to the constraints of different platforms.

Keywords: *post-quantum cryptography, software, software architecture, software system, elliptic curve cryptography, C4 model.*

Дата першого надходження статті до видання: 23.03.2026
Дата прийняття статті до друку після рецензування: 20.04.2026
Дата публікації (оприлюднення) статті: 19.05.2026