УДК 004.056.55:004.42

М. В. Онай, Ф. В. Петренко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Проспект Берестейський, 37, 03056 Київ, Україна

Аналіз продуктивності постквантових криптосистем Національного інституту стандартів і технологій у середовищі .NET

Представлено результати експериментального аналізу продуктивності вбудованих реалізацій постквантових алгоритмів стандартів NIST (The National Institute of Standards and Technology) FIPS 203 (ML-KEM) і FIPS 204 (ML-DSA) у середовищі .NET 10. За допомогою BenchmarkDotNet визначено швидкодію операцій і показники використання пам'яті на архітектурі Intel Core Ultra. Емпірично підтверджено, що ML-KEM-768 пришвидшує процес генерації ключів більше ніж у 7 разів порівняно з ECC-P256, водночас демонструючи безпрецедентну швидкодію операцій інкапсуляції (майже у 90 разів) і декапсуляції (у понад 50 разів). Встановлено також, що використання PQC (Post-Quantum Cryptography) збільшує розмір криптографічних ключів (зокрема, цифрових підписів) до 34 разів. Це зумовлює ризики гарантованої фрагментації IP-пакетів і вимагає перегляду архітектурних підходів до проєктування в розподілених системах.

Ключові слова: *криптографія, постквантова криптографія, еліптична криптографія, програмне забезпечення, інженерія програмного забезпечення, FIPS 203, FIPS 204, ML-KEM, ML-DSA, .NET 10, BenchmarkDotNet, продуктивність, розподілені системи.*

Вступ

Швидкий розвиток квантових обчислень становить серйозну загрозу для безпеки інформаційних систем. Класичні алгоритми асиметричного шифрування, такі як RSA (Rivest-Shamir-Adleman) і ECC (Elliptic-curve cryptography), ґрунтуються на складності факторизації великих цілих чисел і дискретного логарифмування. Алгоритм Шора, запущений на досить потужному квантовому комп'ютері, може вирішити ці задачі за поліноміальний час, що призводить до повної втрати криптографічного захисту сучасних протоколів (TLS, SSH, VPN). Хоча такі квантові комп'ютери ще досліджуються глобальними технологічними компаніями, стратегія злов-

мисників «збирай зараз, розшифровуй пізніше» (Harvest Now, Decrypt Later) вимагає застосування квантово стійких рішень уже сьогодні [1].

Відповіддю на цей виклик стала ініціатива Національного інституту стандартів і технологій США (NIST) щодо створення стандартів алгоритмів постквантової криптографії (PQC). У 2024 році було прийнято фінальні стандарти FIPS 203 (ML-KEM) для інкапсуляції ключів і FIPS 204 (ML-DSA) для цифрового підпису [2, 3]. Ці алгоритми, засновані на криптографії ґраток, забезпечують захист як проти класичного, так і квантового криптоаналізу. Проте теоретична стійкість не гарантує ефективність у практичних умовах реалізації, особливо в середовищах з керованим кодом, таких як Microsoft .NET.

Актуальність цього дослідження зумовлена недостатньою кількістю емпіричних даних про поведінку стандартизованих алгоритмів PQC у корпоративних екосистемах. Більшість існуючих бенчмарків зосереджені на реалізації із використанням низькорівневих мов програмування C/C++ або Python, тоді як сектор корпоративного розроблення значною мірою використовує платформу .NET. Поява вбудованої підтримки PQC у .NET 10 та оновленнях Windows 11 CNG відкриває нові можливості для інтеграції, але одночасно піднімає важливі питання щодо продуктивності JIT-компіляції, ефективності Garbage Collector при роботі з великими криптографічними об’єктами.

Метою цієї роботи є аналіз продуктивності вбудованих реалізацій постквантових алгоритмів ML-KEM та ML-DSA, а також порівняння з іншими класичними алгоритмами у середовищі .NET. Для забезпечення точності вимірювань використовується інструментарій BenchmarkDotNet [4]. Результати роботи, доступні у репозиторії [5], мають за мету допомогти архітекторам програмного забезпечення при прийнятті рішень щодо переходу на постквантову криптографію.

Огляд останніх досліджень і публікацій

Аналіз досліджень і публікацій варто розпочати з фундаментальних специфікацій FIPS 203 та FIPS 204 [2, 3], які формують теоретичний базис постквантової міграції. Хоча ці документи регламентують алгебраїчну структуру алгоритмів: параметри ґраток і вимоги до криптостійкості, вони розглядають криптографічні примітиви як абстрактні математичні функції. Специфікації не містять архітектурних рекомендацій щодо їхньої безпечної реалізації у середовищах з автоматичним керуванням пам’яттю. Це створює концептуальний розрив між теоретичною моделлю безпеки та практичними викликами інтеграції, такими як контроль часу виконання під впливом JIT-компіляції та оптимізації виділення пам’яті у купі (heap allocations), що і зумовлює необхідність подальших прикладних досліджень.

Значна увага приділяється інтеграції цих стандартів у корпоративні платформи в публікаціях Microsoft, зокрема Чарлі Белла [1]. Наголошується на невідкладності підготовки інфраструктури через загрозу ретроспективного розшифрування даних. Технічні аспекти впровадження PQC API в екосистему Windows і Azure описані в роботі [6], де автори демонструють готовність хмарних сервісів до гібридних схем шифрування. Окрім того, дослідження Microsoft Quantum [7, 8] зосереджені на оцінці ресурсів, необхідних для криптоаналізу, що дозволяє теоретично обґрунтувати вибір параметрів безпеки для стандартів FIPS 203/204.

У роботі [9] автори розширюють дослідну основу, розглядаючи практичні аспекти розгортання стандартів FIPS 203 та FIPS 204 у телекомунікаційних мережах. Дослідження підтверджує, що завдяки використанню векторизації AVX2, ці алгоритми досягають конкурентного часу виконання порівняно з класичними схемами. Водночас наголошується, що перехід до PQC у масштабній інфраструктурі (зокрема 5G та 6G) створює виклики, що пов'язані з регуляторною відповідністю та необхідністю впровадження стратегій поетапної міграції. Особлива увага приділяється концепції криптографічної гнучкості (crypto-agility) як ключового інструменту для забезпечення стійкості систем в умовах невизначеності темпів розвитку квантових обчислень.

У дослідженні [10] автори надають комплексні рекомендації щодо впровадження PQC у різних сценаріях. Експериментально доведено, що для пристроїв з обмеженими ресурсами (IoT/Edge) найбільш ефективним механізмом інкапсуляції є ML-KEM завдяки оптимальному балансу швидкості та використання пам'яті. Водночас для серверних та хмарних середовищ швидкодія операцій будь-якого з протестованих алгоритмів NIST, Dilithium та Kyber, не є критичними, оскільки час виконання операцій на сучасному обладнанні становить менше однієї мілісекунди.

Питання інтеграції PQC у сегмент електронних комунікацій досліджено у праці [11]. Автори фокусуються на порівняльному аналізі стандартів CRYSTALS-Dilithium, Falcon та SPHINCS+ при їхньому впровадженні у класичні алгоритми електронного підпису. Зроблено висновок, що висока швидкість створення підпису алгоритмами на базі ґраток є їхньою ключовою перевагою, проте вибір конкретного стандарту (наприклад, між Dilithium та SPHINCS+) має засновуватися на компромісі між шкидкодією, відмовостійкістю та розміром криптографічного ключа.

Додатково, комплексний огляд викликів інформаційної безпеки в умовах квантової загрози наведено у дослідженні [12]. Автори аналізують перспективи розвитку квантово-стійких алгоритмів і наголошують на необхідності перегляду підходів до проєктування складних інформаційних систем. Робота підкреслює, що міграція до PQC — це не лише зміна математичного апарату, а й необхідність забезпечення сумісності з існуючими протоколами в умовах обмежених ресурсів.

Починаючи з 2024 року, дослідження із використанням мов програмування Java/C# ґрунтувалися на сторонніх бібліотеках, таких як Bouncy Castle [13]. Однак із появою вбудованої підтримки у .NET 10 виникає необхідність оцінити саме системні реалізації.

Середовище дослідження

Експериментальні дослідження було проведено на високопродуктивній робочій станції. Вибір центрального процесора Intel Core Ultra 9 обґрунтований підтримкою розширених інструкцій AVX2/VNNI, які суттєво пришвидшують виконання поліноміальних операцій у алгоритмах, заснованих на ґратках.

Ключовою частиною є використання Windows 11 версії 25H2, яка має оновлений стек Cryptography Next Generation (CNG) з покращеною підтримкою постквантових алгоритмів на рівні ядра ОС. Це забезпечує можливість запуску бенчмарків вбудованих реалізацій без додаткових накладних витрат на емуляцію. Детальна інформація про тестове середовище наведена в таблиці.

Характеристики середовища

Компонент	Характеристика
Процесор (CPU)	Intel® Core™ Ultra 9 285H
Тактова частота	2.90 GHz (Base), до 5.4 GHz (Turbo)
Оперативна пам'ять (RAM)	64.0 GB DDR5 (63.6 GB usable)
Операційна система	Windows 11 Pro (25H2, Build 26200.7628)
Бібліотека	System.Security.Cryptography

Предмет дослідження

Дослідження включає порівняльний аналіз двох типів криптографічних примітивів.

1. Механізми інкапсуляції ключів: постквантовий алгоритм ML-KEM-768 (з рівнем безпеки NIST 3, еквівалент AES-192) із класичними алгоритмами RSA-4096 та ECC-P256 (NIST P-256).

2. Цифрові підписи: постквантовий алгоритм ML-DSA-65 (Dilithium, рівень безпеки 3) із RSA-4096 та ECDSA (P-384, що відповідає рівню безпеки PQC-алгоритму).

Методологія та метрики

Бібліотека BenchmarkDotNet допомагає мінімізувати вплив «холодного старту» (Cold Start) і системного шуму за допомогою серії ітерацій (Warmup + Target) [6]. Для комплексної оцінки ефективності алгоритмів у середовищі було обрано наступний набір метрик.

Середній час виконання: основний показник обчислювальної складності, що визначає затримку при виконанні криптографічних операцій.

Виділена пам’ять: обсяг пам’яті, що виділяється в купі за одну криптографічну операцію. Ця метрика є критично важливою для керованого середовища .NET, оскільки надмірне навантаження призводить до збільшення частоти викликів Garbage Collector (GC). Часті збірки сміття створюють мікропаузи в роботі додатка, що експоненційно знижує загальну пропускну здатність і масштабованість високонавантажених серверних систем.

Розмір відкритих ключів, шифротекстів і цифрових підписів у байтах. Враховуючи значні обсяги даних у PQC, цей показник безпосередньо впливає на фрагментацію IP-пакетів та ефективність використання пропускної здатності мережі.

Аналіз результатів експериментів

Результати бенчмаркінгу, проведеного на платформі .NET 10 із використанням бібліотеки System.Security.Cryptography та інструкцій процесора AVX2, демонструють фундаментальні відмінності в продуктивності між класичними алгоритмами (RSA, ECC) та новими постквантовими стандартами. Аналіз отриманих даних дозволяє виділити чотири ключові вектори впливу PQC на архітектуру програмного забезпечення.

1. Швидкодія операцій алгоритмів інкапсуляції.

На основі отриманих даних можна зробити висновок, що перехід на постквантову криптографію ґраток (Lattice-based cryptography) не створює критичного навантаження на центральний процесор.

Як видно на рис. 1, особливу увагу привертає операція генерації ключів. Алгоритм ML-KEM-768 продемонстрував середній час виконання на рівні 58 мкс, що значно швидше за генерацію ключів на еліптичних кривих (ECDH-P256 ~414 мкс) і на кілька порядків швидше за RSA-4096. Це пояснюється природою математичних операцій: ML-KEM побудований на основі операцій над поліномами у скінченних полях, які ефективно розпаралелюються сучасними процесорами, на відміну від складної арифметики великих чисел у класичних алгоритмах.

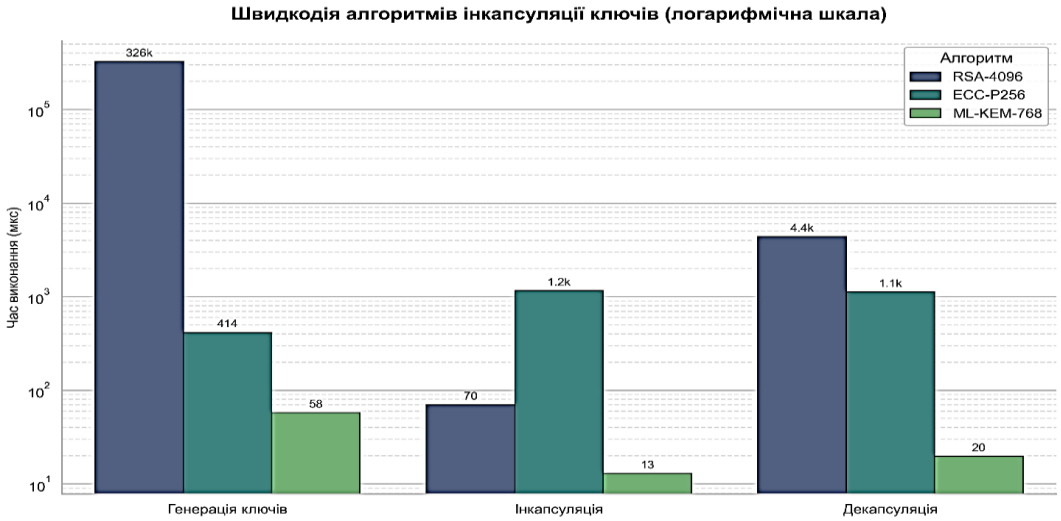

Рис. 1. Порівняльна діаграма часу виконання операцій інкапсуляції ключів у логарифмічному масштабі

Операція інкапсуляції, яка є найбільш частою у сценаріях встановлення TLS з'єднання, демонструє беззаперечну перевагу постквантового стандарту. Для ML-KEM-768 вона виконується всього за 12,85 мкс, що майже в 6 разів швидше за RSA-4096 (~70 мкс) та у 90 разів швидше за вбудовану реалізацію ECC-P256 (~1200 мкс). Такі високі обчислювальні показники повністю спростовують побоювання щодо критичного зниження продуктивності процесорів серверів.

Не менш критичною для масштабування високонавантажених серверів є операція декапсуляції, яка виконується на стороні отримувача для відновлення спільного секрету. Відповідно до результатів тесту, ML-KEM-768 здійснює декапсуляцію за ~20 мкс. Для порівняння, архітектурно правильна декапсуляція за допомогою ECC-P256 потребує ~1100 мкс (повільніше у 55 разів), а розшифрування секрету алгоритмом RSA-4096 займає ~4400 мкс. Такі екстремально високі обчислювальні показники постквантового алгоритму повністю спростовують побоювання щодо зниження продуктивності на боці сервера.

2. Швидкодія операцій цифрових підписів.

Аналіз алгоритмів цифрового підпису виявив іншу динаміку порівняно з механізмами інкапсуляції.

Алгоритм ML-DSA-65 (Dilithium) демонструє виражену асиметрію навантаження (рис. 2):

— генерація ключів: додатковою перевагою стандарту є висока швидкість генерації ключових пар (~564 мкс). У досліджуваному середовищі ML-DSA-65 виявився навіть швидшим за класичний алгоритм на еліптичних кривих ECDSA-P384 (~726 мкс) і на три порядки випередив повільну генерацію простих чисел в RSA-4096 (~482 мс);

— створення підпису: середній час виконання становить ~481 мкс. Хоча цей показник у 3 рази перевищує час роботи еліптичних кривих (ECDSA-P384 виконується за ~161 мкс), він демонструє більш ніж п'ятикратну перевагу над класичним RSA-4096, якому для аналогічної операції потрібно понад 2472 мкс. Таке пришвидшення дозволяє суттєво розвантажити сервери центрів сертифікації (CA) при масовій генерації токенів чи підписуванні документів;

— перевірка підпису: результат у ~39 мкс свідчить про безпрецедентно високу ефективність верифікації. Показово, що в досліджуваному керованому середовищі .NET алгоритм ML-DSA перевершив навіть еталонні показники RSA-4096 (~71 мкс), не кажучи вже про ECDSA (~166 мкс). Це робить ML-DSA ідеальним кандидатом для використання на клієнтських пристроях (Edge/IoT) з обмеженими обчислювальними ресурсами, де операція перевірки виконується найчастіше.

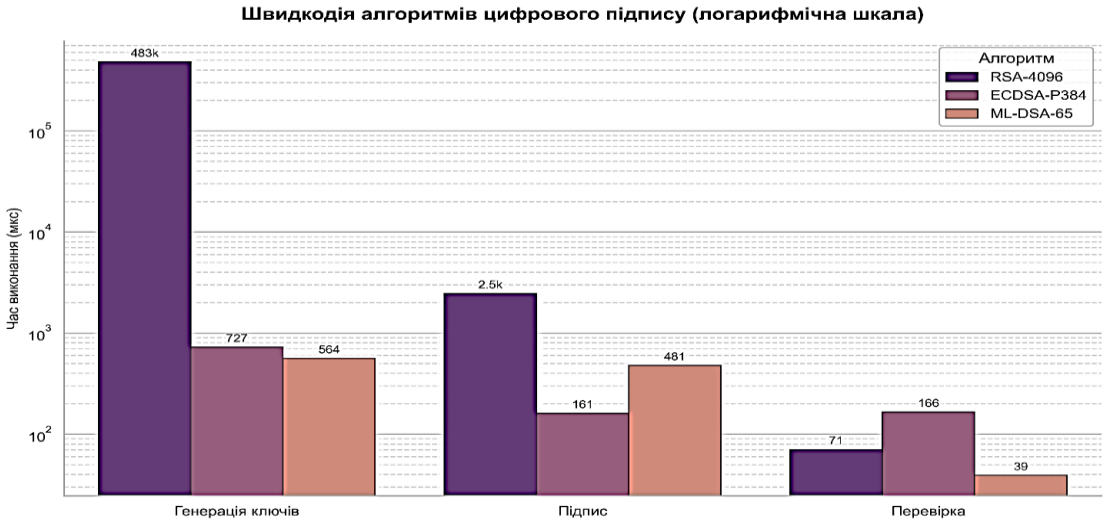

Рис. 2. Порівняння середнього часу виконання операцій створення та перевірки цифрового підпису для алгоритмів RSA, ECDSA та ML-DSA

Отже, результати експерименту підтверджують, що з обчислювальної точки зору перехід на постквантовий стандарт ML-DSA забезпечує експоненційно вищий рівень захисту без втрати, а в багатьох сценаріях, і з суттєвим приростом продуктивності.

3. Аналіз криптографічних об'єктів і їхній вплив на фрагментацію пакетів.

Найбільш суттєвим викликом, виявленим у ході дослідження, є розмір криптографічних об'єктів.

Діаграма на рис. 3 ілюструє проблему перевищення MTU:

— цифровий підпис: розмір підпису ML-DSA-65 становить 3309 байт, що у 2.2 рази перевищує стандартний пакет Ethernet (1500 байт). Передача одного підпису призводить до фрагментації IP-пакета, що збільшує ризик втрати даних у протоколах UDP/QUIC;

— механізм інкапсуляції ключів: публічний ключ ML-KEM-768 (1206 байт) хоч і вписується в один пакет, але у 13 разів перевищує розмір ключа ECC-P256 (91 байт).

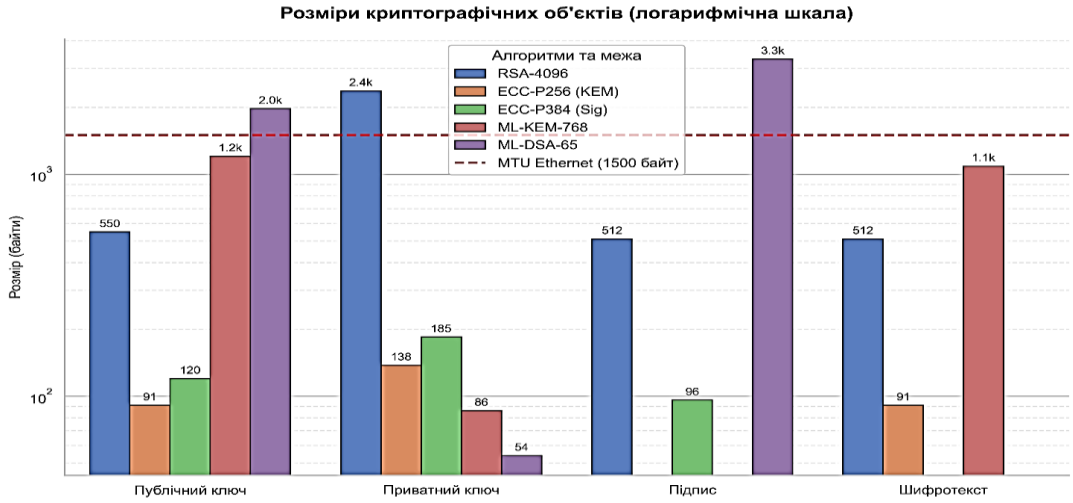

Рис. 3. Порівняння розмірів криптографічних об’єктів
і їхнє співвідношення з MTU Ethernet (1500 байт)

Під час аналізу важливо враховувати архітектурні відмінності алгоритмів. У класичному протоколі узгодження ключів на основі еліптичних кривих роль інкапсульованого шифротексту виконує ефемерний відкритий ключ (розміром 91 байт для кривої P-256), який і передається каналом зв’язку. Натомість постквантовий алгоритм ML-KEM-768 під час операції інкапсуляції генерує повноцінний криптографічний шифротекст обсягом 1088 байтів. Отже, порівняно з класичними еліптичними кривими, міграція на PQC призводить до збільшення транзитного мережевого навантаження на етапі узгодження спільного секрету майже у 12 разів.

Також, експериментально зафіксовано аномалію розміру приватних ключів: об’єкт приватного ключа ML-DSA-65 у пам’яті займає лише 54 байти (проти теоретичних кількох кілобайт). Це підтверджує, що реалізація Windows CNG зберігає лише криптографічне «зерно» (seed), розгортаючи повні матриці динамічно [3].

4. Виділення пам’яті при криптографічних операціях.

Дослідження використання оперативної пам’яті підтвердило високу якість реалізації PQC у .NET 10.

Показник виділення пам’яті для операції інкапсуляції алгоритмом ML-KEM-768 (рис. 4) становить лише 1200 байтів, що фактично дорівнює сумарному розміру вихідних даних (шифротексту обсягом 1088 байтів та 32-байтного спільного секрету). Це свідчить про відсутність прихованих виділень пам’яті в купі та відсутність надлишкового тиску на систему збирання сміття.

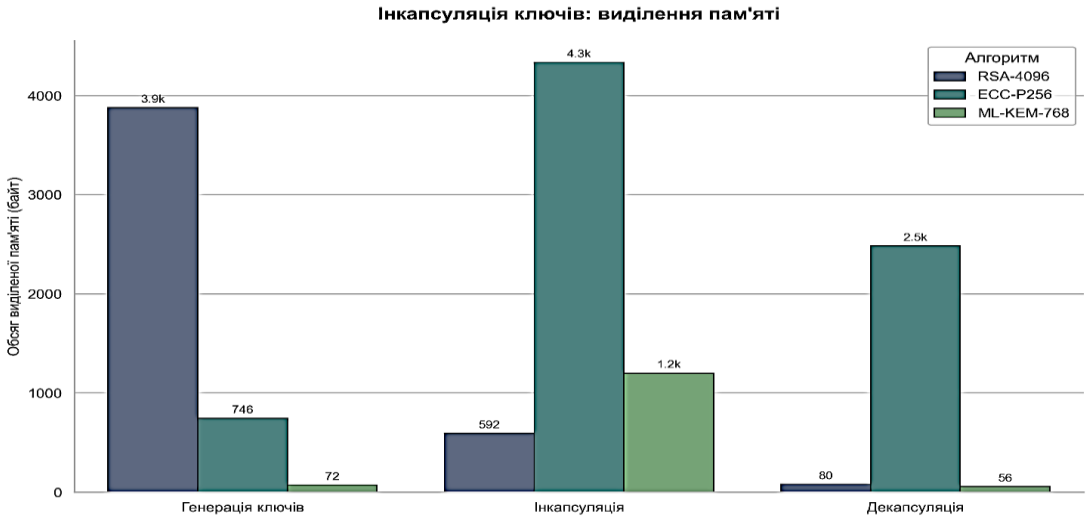

Рис. 4. Показники виділення динамічної пам’яті при виконанні операцій інкапсуляції

Аналіз програмної реалізації класичного алгоритму ECC-P256 виявив архітектурні обмеження стандартних криптографічних провайдерів у платформі .NET. Так, під час операції декапсуляції, яка математично вимагає лише обчислення скалярного добутку статичного приватного ключа і отриманого шифротексту, абстракція API змушує ініціалізувати новий об’єкт криптопровайдера. Це призводить до генерації непотрібної ключової пари, що штучно завищує процесорний час виконання декапсуляції у класичних системах і додатково виділяє 2484 байти пам’яті за кожну транзакцію. Натомість API постквантового алгоритму ML-KEM позбавлене таких архітектурних недоліків, що робить його інтеграцію у високонавантажені розподілені системи значно ефективнішою з точки зору використання CPU.

Напрямки подальших досліджень

Отримані результати формують аналітичну основу для наступних етапів дослідження. Рекомендується зосередити роботу на розробленні методів проєктування квантово стійких систем на основі мікросервісної архітектури та архітектури на основі подій (Event-driven architecture). В якості ключових елементів подальших досліджень пропонується: застосування квантово стійких токенів доступу (JWT), динамічна конфігурації криптографічних алгоритмів з поєднанням класичних і постквантових, побудова моделі комунікації між сервісами, яка дозволяє оптимізувати ресурсні витрати та зменшити навантаження на мережу.

Висновки

Підсумовуючи результати дослідження, можна стверджувати, що перехід до постквантової криптографії (PQC) в екосистемі корпоративного програмного забезпечення є не лише заміною математичних примітивів, а й комплексним архітектурним викликом. Експериментальний аналіз реалізованих стандартів FIPS 203 (ML-KEM) та FIPS 204 (ML-DSA) на платформі .NET 10 виявив суттєву асиметрію між обчислювальними витратами та розміром криптографічних об’єктів.

Емпірично встановлено, що сучасні реалізації механізмів інкапсуляції ключів на основі ґраток є ефективними з точки зору обчислень. Зокрема, алгоритм ML-KEM-768 продемонстрував пришвидшення операції генерації ключів у понад 7 разів порівняно з еліптичними кривими (NIST P-256), водночас забезпечуючи безпрецедентні показники операцій інкапсуляції (майже у 90 разів) і декапсуляції (у понад 50 разів). Це свідчить про те, що впровадження квантової стійкості не призведе до зниження швидкодії процесорів серверів.

Однак критичним обмежуючим фактором стає розмір криптографічних об'єктів. Дослідження зафіксувало збільшення обсягу цифрових підписів до 34 разів (для ML-DSA-65 порівняно з ECDSA-P384). Оскільки розмір підпису (~3300 байт) суттєво перевищує стандартний MTU Ethernet, це створює умови гарантованої фрагментації пакетів, що вимагає перегляду підходів до проєктування протоколів на основі UDP/QUIC і механізмів буферизації у високонавантажених системах.

Під час тестування виявлено ефективну стратегію керування пам'яттю, де розмір об'єктів приватних ключів PQC мінімізовано (54–86 байт) за рахунок зберігання лише криптографічного зерна та динамічної регенерації матриць. Це дозволяє використовувати алгоритми у системах з обмеженими ресурсами пам'яті без ризику переповнення купи (Heap).

З точки зору інженерії програмного забезпечення, отримані результати обґрунтовують доцільність переходу до гібридних схем криптографічного захисту. Це дозволить використовувати переваги постквантових алгоритмів, особливо для високонавантажених програмних систем, з перевіреною надійністю класичних алгоритмів.

1. Bell C. Building a quantum-safe future. Microsoft Official Blog. 2023. May 31. URL: https://blogs.microsoft.com/blog/2023/05/31/building-a-quantum-safe-future/.

2. National Institute of Standards and Technology. Module-Lattice-Based Key-Encapsulation Mechanism Standard. Federal Information Processing Standards Publication (FIPS) 203. Gaithersburg, MD: NIST, 2024. URL: https://csrc.nist.gov/pubs/fips/203/final

3. National Institute of Standards and Technology. Module-Lattice-Based Digital Signature Standard. Federal Information Processing Standards Publication (FIPS) 204. Gaithersburg, MD: NIST, 2024. URL: https://csrc.nist.gov/pubs/fips/204/final.

4. BenchmarkDotNet. The .NET Foundation. URL: https://benchmarkdotnet.org/.

5. Petrenko F. PQC Research App: Source code for benchmarking NIST FIPS 203/204 algorithms in .NET. GitHub repository. 2026. URL: https://github.com/FedyaPetrenko/pqc-research.

6. Microsoft Security Team. Post-Quantum Cryptography APIs Now Generally Available on Microsoft Platforms. Microsoft Security Blog. 2025. URL: https://techcommunity.microsoft.com/blog/microsoft-security-blog/post-quantum-cryptography-apis-now-generally-available-on-microsoft-platforms/4469093.

7. Quantum Resource Estimation and Cryptography. Microsoft Azure Quantum. URL: https://quantum.microsoft.com/en-us/tools/quantum-cryptography.

8. Calculating resource estimates for cryptanalysis. Microsoft Azure Quantum Blog. URL: https://quantum.microsoft.com/en-us/insights/blogs/resource-estimation/calculating-resource-estimates-for-cryptanalysis

9. Performance Analysis and Industry Deployment of Post-Quantum Cryptography Algorithms. arXiv preprint arXiv:2503.12952. 2025. URL: https://arxiv.org/abs/2503.12952.

10. A Practical Performance Benchmark of Post-Quantum Cryptography Across Heterogeneous Computing Environments. Computers. 2020. Vol. 9, No. 2. P. 32. URL: https://www.mdpi.com/2410-387X/9/2/32..

11. Фесенко А., Мирошніченко М. Порівняння постквантових стандартів у розрізі впровадження у класичні алгоритми електронного підпису. *Безпека інформаційних систем і технологій*. 2024. № 1 (7). С. 31–38. DOI: https://doi.org/10.17721/ISTS.2024.7.31-38.

12. Петрушко І., Поліщук В., Матей А. Постквантова криптографія: виклики та перспективи розробки квантово-стійких алгоритмів для забезпечення безпеки інформаційних систем. *Вісник Хмельницького національного університету. Серія: Технічні науки*. 2025. № 347 (1). С. 461–467. DOI: https://doi.org/10.31891/2307-5732-2025-347-63.

13. Bouncy Castle Cryptography Library for .NET. The Legion of the Bouncy Castle. URL: https://www.bouncycastle.org/download/bouncy-castle-c/.

Надійшла до редакції 22.02.2026