Міністерство освіти і науки України
Харківського національного університету імені В.Н. Каразіна
Навчально-наукового інституту комп'ютерних наук та штучного інтелекту
Спеціальність 125 «Кібербезпека та захист інформації»
Освітня програма «Безпека інформаційних і комунікаційних систем»

В.о. зав. кафедрою КІСМТ
Марина ЄСІНА
«Допущено до захисту»

“ “ _____2024р.

Пояснювальна записка

до кваліфікаційної роботи магістра

на тему: «Обґрунтування вибору, дослідження та програмна модель кандидата на квантово стійкий міжнародний електронний підпис (ЕП) KAZ-SIGN»

оцінка “_____”

Голова ЕК
Лемешко О.В.

Керівник роботи: д. т. н., проф. Горбенко І. Д.

Рецензент: к.т.н. Голубничий Д.Ю.

Виконавець: студент групи КБ-61
Савченко Ігор Євгенович

Харків 2024

РЕФЕРАТ

Кваліфікаційна робота: 62 сторінок, 19 рисунків, 2 таблиць, 1 додаток, 22 джерела.

Метою роботи є теоретичне обґрунтування, дослідження та розробка програмної моделі квантово стійкого електронного підпису KAZ-SIGN, а також оцінка його характеристик та порівняння з існуючими аналогами.

У роботі використано методи теоретичного аналізу, математичного та комп'ютерного моделювання, порівняльного аналізу. Дослідження проводилися з використанням сучасних програмних засобів розробки та тестування криптографічних алгоритмів, включаючи спеціалізовані бібліотеки для роботи з постквантовими примітивами.

В результаті роботи розроблено та реалізовано програмну модель алгоритму KAZ-SIGN, проведено всебічне дослідження його криптографічної стійкості та експлуатаційних характеристик. Новизна роботи полягає в удосконаленні методу оцінки криптографічної стійкості електронного підпису до квантових атак та оптимізації параметрів алгоритму для підвищення його ефективності.

Результати роботи рекомендується використовувати при розробці та впровадженні систем електронного підпису, стійких до атак з використанням квантових комп'ютерів. Програмна реалізація може бути інтегрована в існуючі криптографічні бібліотеки та використана як основа для подальших досліджень.

Практична значущість роботи полягає в можливості використання розробленої програмної моделі для оцінки характеристик та тестування постквантових алгоритмів електронного підпису. Результати порівняльного аналізу дозволяють обґрунтовано вибирати алгоритми для конкретних застосувань.

Подальші дослідження можуть бути спрямовані на оптимізацію параметрів алгоритму для різних платформ, розробку апаратних реалізацій, а також на дослідження можливостей інтеграції з існуючими криптографічними протоколами та системами.

Ключові слова: ПОСТКВАНТОВА КРИПТОГРАФІЯ, ЕЛЕКТРОННИЙ ПІДПИС, KAZ-SIGN, КРИПТОГРАФІЧНА СТІЙКІСТЬ, КВАНТОВИЙ КОМП'ЮТЕР, КРИПТОГРАФІЧНІ ПРИМІТИВИ, ПРОГРАМНА МОДЕЛЬ, МАТЕМАТИЧНИЙ АПАРАТ, КРИПТОАНАЛІЗ, ОПТИМІЗАЦІЯ ПАРАМЕТРІВ.

ABSTRACT

Qualification work: 62 pages, 19 figures, 2 tables, 1 appendix, 22 references.

The aim of the work is theoretical justification, research, and development of a software model for the quantum-resistant digital signature KAZ-SIGN, as well as evaluation of its characteristics and comparison with existing alternatives.

The research employs methods of theoretical analysis, mathematical and computer modeling, and comparative analysis. The studies were conducted using modern software development and testing tools for cryptographic algorithms, including specialized libraries for working with post-quantum primitives.

As a result of the work, a software model of the KAZ-SIGN algorithm was developed and implemented, and a comprehensive study of its cryptographic security and operational characteristics was conducted. The novelty of the work lies in improving the method of evaluating the cryptographic resistance of digital signatures to quantum attacks and optimizing algorithm parameters to increase its efficiency.

The results of the work are recommended for use in the development and implementation of digital signature systems resistant to quantum computer attacks. The software implementation can be integrated into existing cryptographic libraries and used as a basis for further research.

The practical significance of the work lies in the possibility of using the developed software model for evaluating characteristics and testing post-quantum digital signature algorithms. The results of the comparative analysis allow for informed selection of algorithms for specific applications.

Further research may be directed towards optimizing algorithm parameters for different platforms, developing hardware implementations, and investigating possibilities for integration with existing cryptographic protocols and systems.

Keywords: POST-QUANTUM CRYPTOGRAPHY, DIGITAL SIGNATURE, KAZ-SIGN, CRYPTOGRAPHIC SECURITY, QUANTUM COMPUTER, CRYPTOGRAPHIC PRIMITIVES, SOFTWARE MODEL, MATHEMATICAL APPARATUS, CRYPTANALYSIS, PARAMETER OPTIMIZATION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АЕП - асиметричний електронний підпис

ВК - відкритий ключ

ГК - генерація ключів

ЕП - електронний підпис

ЗК - закритий ключ

КС - криптографічна система

НСКЗ - національна система конфіденційного зв'язку

ПК - приватний ключ

ПКІ - публічна ключова інфраструктура

ЦП - цифровий підпис

ECDSA - Elliptic Curve Digital Signature Algorithm (алгоритм цифрового підпису на еліптичних кривих)

ETSI - European Telecommunications Standards Institute (Європейський інститут телекомунікаційних стандартів)

ISO - International Organization for Standardization (Міжнародна організація зі стандартизації)

KAZ-SIGN - Kazakhstan Signature (казахстанський алгоритм цифрового підпису)

NIST - National Institute of Standards and Technology (Національний інститут стандартів і технологій США)

PQC - Post-Quantum Cryptography (постквантова криптографія)

RSA - Rivest-Shamir-Adleman (криптографічний алгоритм з відкритим ключем)

SHA - Secure Hash Algorithm (алгоритм безпечного хешування)

XOF - eXtendable Output Function (функція з розширюваним виходом)

ВСТУП

Розвиток квантових комп'ютерів створює значну загрозу для існуючих криптографічних систем, особливо для алгоритмів електронного підпису, що базуються на проблемах факторизації та дискретного логарифмування. Провідні міжнародні організації, такі як NIST, ETSI та ISO, активно працюють над стандартизацією постквантових криптографічних алгоритмів. Аналіз останніх досліджень демонструє, що найперспективнішими напрямками є підписи на основі решіток, геш-функцій та мультиваріативних перетворень. Серед провідних розробників можна виділити такі компанії як IBM, Google та Microsoft, які інвестують значні ресурси в дослідження постквантових алгоритмів.

Актуальність даної роботи обумовлена необхідністю захисту інформації від атак з використанням квантових комп'ютерів. Існує нагальна потреба в розробці та дослідженні нових алгоритмів електронного підпису, які будуть стійкими до квантових атак. Особливої уваги заслуговує дослідження та оцінка нових кандидатів на постквантові стандарти, серед яких KAZ-SIGN розглядається як перспективний кандидат на роль квантово стійкого електронного підпису.

Метою роботи є теоретичне обґрунтування, дослідження та розробка програмної моделі квантово стійкого електронного підпису KAZ-SIGN. Для досягнення поставленої мети необхідно вирішити наступні завдання: провести аналіз сучасного стану розробки квантово стійких ЕП; дослідити вимоги NIST до постквантових криптографічних примітивів; обґрунтувати вибір та дослідити метод KAZ-SIGN; розробити програмну модель та оцінити характеристики KAZ-SIGN.

Об'єктом дослідження є процеси формування та перевірки електронного цифрового підпису в умовах постквантової криптографії. Предметом

дослідження виступають методи та алгоритми квантово стійкого електронного підпису KAZ-SIGN. В роботі використовуються методи теоретичного аналізу, математичного та комп'ютерного моделювання, а також порівняльного аналізу.

Наукова новизна роботи полягає в удосконаленні методу оцінки криптографічної стійкості ЕП до квантових атак, розробці програмної моделі KAZ-SIGN з оптимізованими параметрами та отриманні нових експериментальних даних щодо характеристик KAZ-SIGN.

Практичне значення отриманих результатів визначається можливістю використання розробленої програмної моделі для подальших досліджень. Результати роботи можуть бути застосовані при розробці стандартів, а отримані характеристики дозволяють оцінити перспективність впровадження KAZ-SIGN в реальні системи.

Дане дослідження проводиться в рамках міжнародного проекту з розробки постквантових криптографічних примітивів та тісно пов'язане з дослідженнями NIST PQC. Отримані результати доповнюють існуючі дослідження в області постквантової криптографії та можуть бути використані для подальшого розвитку цього напрямку.

1 АНАЛІЗ СУЧАСНОГО СТАНУ РОЗВИТКУ КВАНТОВО СТІЙКИХ ЕЛЕКТРОННИХ ПІДПИСІВ

1.1 Аналіз міжнародних стандартів та проектів квантово стійких ЕП

1.1.1 Історія розвитку постквантової криптографії

Історія розвитку постквантової криптографії бере свій початок у 1994 році, коли Пітер Шор запропонував квантовий алгоритм, здатний ефективно розв'язувати задачі факторизації та дискретного логарифмування. Це відкриття стало революційним, оскільки показало потенційну вразливість більшості існуючих криптосистем з відкритим ключем до атак з використанням квантового комп'ютера. Після цього криптографічна спільнота почала активно досліджувати альтернативні підходи до криптографії, які б залишались безпечними навіть при наявності потужних квантових комп'ютерів [1].

Період з 1994 по 2000 роки характеризується першими теоретичними розробками в області постквантової криптографії. У 1996 році Лов Гровер представив квантовий алгоритм пошуку, який додатково підкреслив необхідність розробки нових криптографічних примітивів. До кінця 90-х років з'явились перші пропозиції щодо криптосистем, які могли б протистояти атакам з використанням квантових комп'ютерів.

Протягом 2001-2010 років відбувалось формування основних напрямків постквантової криптографії. У 2003 році були розроблені перші криптосистеми на решітках, а в 2005 році розпочались активні дослідження в області мультиваріативної криптографії. Важливою віхою стала поява самого терміну «Post-Quantum Cryptography» у 2008 році, а проведення першої міжнародної конференції PQCrypto у 2009 році закріпило цей напрямок як окрему галузь криптографії [2].

Період 2011-2015 років відзначився значною активізацією досліджень у цій області. Викриття Едварда Сноудена у 2013 році значно підвищило інтерес до квантово-стійкої криптографії, а в 2015 році NSA офіційно оголосило про плани переходу на постквантові алгоритми. Це стало потужним стимулом для подальших досліджень та розробок.

Особливо важливим етапом став період 2016-2020 років, коли NIST оголосив конкурс постквантових криптографічних алгоритмів. На конкурс було подано 69 кандидатів, з яких до другого раунду у 2019 році пройшли 26, а до третього раунду у 2020 році - 7 фіналістів. Цей конкурс став ключовим елементом у процесі стандартизації постквантових алгоритмів.

Ключові етапи розвитку постквантової криптографії зображені на рисунку 1.1.

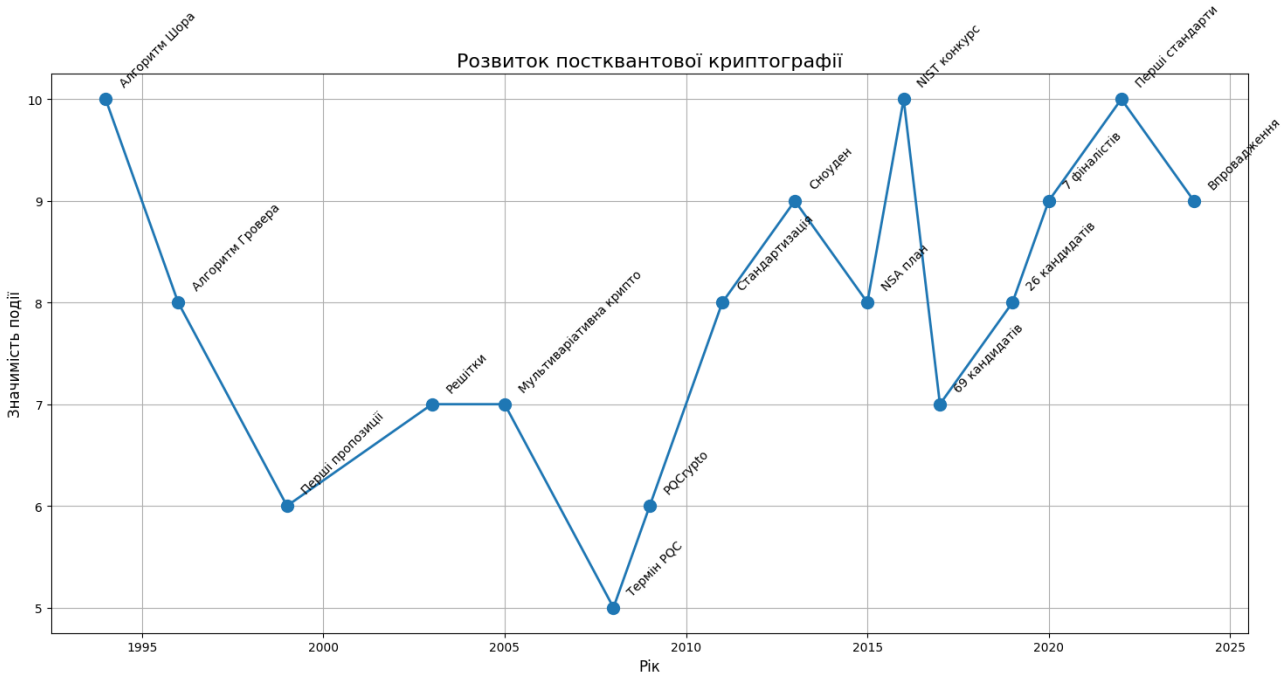

Рисунок 1.1 - Ключові етапи розвитку постквантової криптографії

Сучасний етап (2021-2024) характеризується практичним впровадженням постквантових алгоритмів. У 2022 році NIST обрав перші стандарти постквантової криптографії: CRYSTALS-Kyber для шифрування та CRYSTALS-

Dilithium, FALCON, SPHINCS+ для цифрового підпису. У 2023-2024 роках спостерігається активне впровадження цих алгоритмів у реальні системи та розробка нових кандидатів, включаючи KAZ-SIGN.

На сьогоднішній день основними викликами залишаються великі розміри ключів порівняно з класичними алгоритмами, складність реалізації на обмежених пристроях та необхідність забезпечення сумісності з існуючими системами. Проте активний розвиток цієї галузі та міжнародна підтримка дають підстави очікувати появу ще більш ефективних рішень у найближчому майбутньому [3].

1.1.2 Огляд існуючих міжнародних стандартів

В області постквантової криптографії існує декілька ключових міжнародних стандартів, які визначають основні вимоги та рекомендації щодо розробки та впровадження квантово-стійких криптографічних примітивів.

NIST SP 800-208 «Рекомендації щодо постквантової криптографії»

Цей документ, опублікований Національним інститутом стандартів і технологій США, є одним з основоположних у сфері постквантової криптографії. Основні положення стандарту включають нижче описані аспекти.

Визначення рівнів безпеки для постквантових алгоритмів:

- Рівень 1: відповідає 128-бітній класичній безпеці;
- Рівень 2: відповідає 192-бітній класичній безпеці;
- Рівень 3: відповідає 256-бітній класичній безпеці;
- Рівень 5: найвищий рівень безпеки.

Вимоги до реалізації:

- Специфікації алгоритмів;
- Формати ключів та підписів;
- Процедури генерації параметрів;
- Методи тестування.

ETSI TR 103 616 «Квантово-безпечні криптографічні примітиви».

Європейський інститут телекомунікаційних стандартів (ETSI) розробив технічний звіт, який фокусується на описаних нижче аспектів.

Класифікації постквантових криптографічних примітивів:

- Криптографія на решітках;
- Мультиваріативна криптографія;
- Криптографія на основі геш-функцій;
- Криптографія на основі кодів.

Рекомендації щодо:

- Вибору параметрів;
- Оцінки безпеки;
- Практичної реалізації;
- Інтеграції з існуючими системами.

ISO/IEC 14888 «Цифрові підписи з додатком».

Міжнародний стандарт, який визначає:

- Загальну структуру цифрових підписів;
- Вимоги до механізмів підпису;
- Формати даних та протоколи.

В контексті постквантової криптографії стандарт включає:

- Специфікації для нових типів підписів;
- Вимоги до сумісності;
- Методи верифікації.

Візуалізація взаємозв'язку стандартів показана на рисунку 1.2

Ці стандарти формують комплексну основу для розробки та впровадження постквантових криптографічних рішень, забезпечуючи необхідний рівень безпеки та сумісності в умовах потенційної загрози з боку квантових комп'ютерів.

Рисунок 1.2 – Взаємозв’язок стандартів

1.1.3 Аналіз поточних проектів стандартизації

Конкурс NIST PQC.

Конкурс постквантової криптографії, організований Національним інститутом стандартів і технологій США (NIST), став найважливішою віхою у розвитку постквантової криптографії. Розпочатий у 2016 році, цей конкурс мав на меті визначити та стандартизувати алгоритми, які зможуть протистояти атакам з використанням квантових комп'ютерів.

Процес відбору був ретельним та багатоетапним. На початковому етапі було подано 69 кандидатів, що представляли широкий спектр криптографічних підходів. Кожен алгоритм проходив всебічний аналіз з точки зору безпеки, продуктивності та практичності реалізації. До другого раунду у 2019 році пройшли 26 найбільш перспективних кандидатів, які продемонстрували найкращий баланс між безпекою та ефективністю.

Третій раунд конкурсу, що розпочався у 2020 році, став вирішальним. Сім фіналістів пройшли додаткові випробування та аналіз. У результаті, в липні 2022 року NIST оголосив свій вибір. CRYSTALS-Kyber був обраний для шифрування з відкритим ключем, а для цифрових підписів були стандартизовані CRYSTALS-Dilithium як основний алгоритм, FALCON як альтернативний варіант, та SPHINCS+ як рішення на основі геш-функцій [4].

Європейські ініціативи (PQCrypto).

Європейський проект PQCrypto представляє собою комплексну дослідницьку ініціативу, що об'єднує провідних експертів та організації з різних країн Європейського Союзу.

На відміну від конкурсу NIST, який фокусується на виборі конкретних алгоритмів, європейський підхід більше орієнтований на фундаментальні дослідження та розробку методологічної бази.

Важливим аспектом європейських ініціатив є їх спрямованість на створення цілісної екосистеми постквантової криптографії. Це включає не лише розробку нових алгоритмів, але й дослідження їх практичного застосування в різних сценаріях використання. Особлива увага приділяється питанням сумісності з існуючими системами та розробці стратегій плавного переходу до постквантових рішень.

Результати роботи проекту PQCrypto мають значний вплив на розвиток галузі. Створена бібліотека постквантових алгоритмів та методологія оцінки їх безпеки широко використовуються дослідниками та розробниками. Регулярні конференції та семінари, організовані в рамках проекту, стали важливими майданчиками для обміну досвідом та координації зусиль міжнародної спільноти.

Наприклад процес відбору кандидатів NIST PQC показаний на рисунку 1.3.

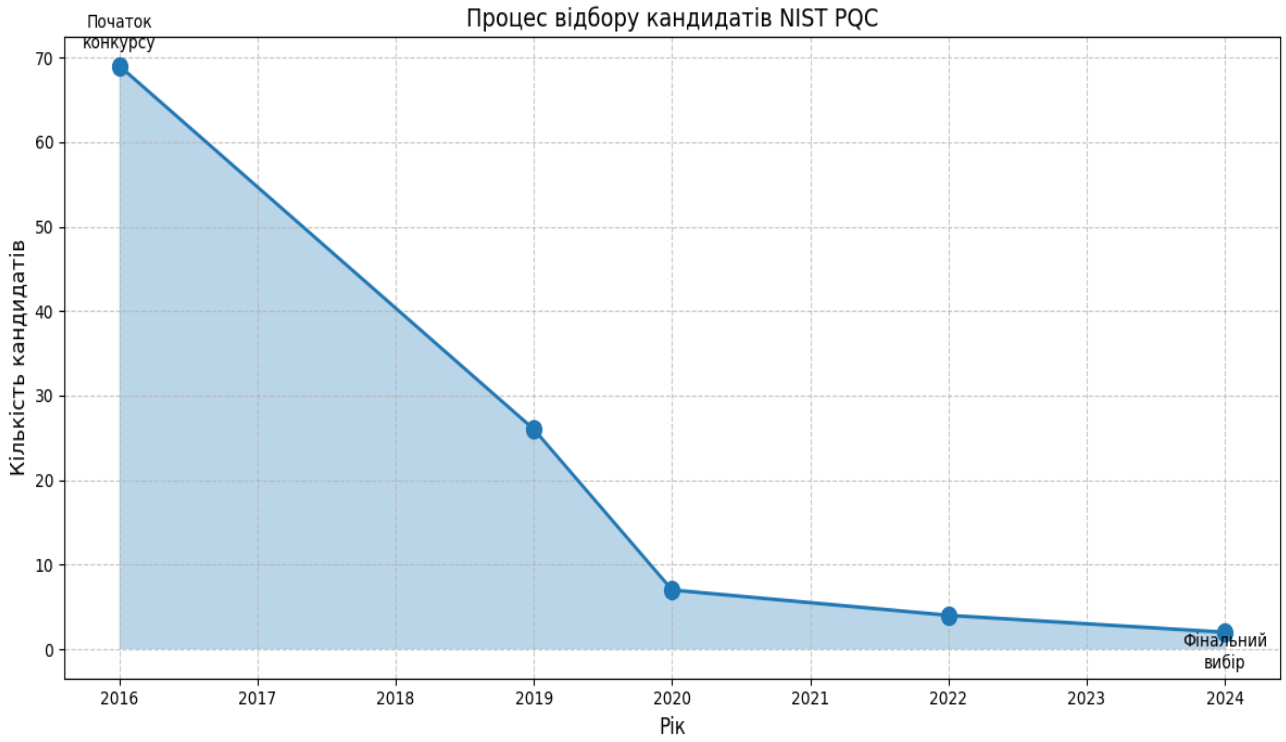

Рисунок 1.3 – Процес відбору кандидатів NIST PQC

Взаємодія між конкурсом NIST PQC та європейськими ініціативами створює синергетичний ефект, що сприяє швидкому розвитку постквантової криптографії. Стандарти, розроблені NIST, доповнюються глибокими теоретичними дослідженнями та практичними рекомендаціями європейських проектів, формуючи комплексний підхід до забезпечення криптографічної безпеки в постквантову еру.

1.1.4 Азійські розробки та основні напрямки розвитку постквантових ЕП

Азійські розробки.

Азійський регіон демонструє значний прогрес у розробці постквантових криптографічних рішень. Особливу увагу привертає розробка KAZ-SIGN, що представляє собою інноваційний підхід до створення квантово-стійкого електронного підпису. Цей алгоритм, розроблений казахстанськими криптографами, базується на математичному апараті алгебраїчних решіток та використовує унікальні властивості локальних кілець.

Китайські дослідники також активно працюють над власними постквантовими рішеннями. Їхні розробки зосереджені на мультиваріативній криптографії та системах на основі решіток. Значний внесок зроблено в оптимізацію існуючих алгоритмів для використання на пристроях з обмеженими ресурсами.

Японські наукові центри фокусуються на розробці гібридних систем, які поєднують класичні та постквантові алгоритми. Їхній підхід характеризується особливою увагою до практичної реалізації та забезпечення сумісності з існуючими криптографічними протоколами.

Південнокорейські розробки відзначаються інноваційними підходами до оптимізації постквантових алгоритмів. Їхні дослідження спрямовані на створення ефективних реалізацій для мобільних та вбудованих систем.

Основні напрямки розвитку постквантових ЕП.

Сучасний розвиток постквантових електронних підписів характеризується кількома ключовими напрямками. Перший напрямок пов'язаний з удосконаленням алгоритмів на основі решіток. Ці системи демонструють хороший баланс між безпекою та ефективністю, що підтверджується їх вибором як стандартів NIST.

Другий важливий напрямок – розвиток підписів на основі геш-функцій. Такі системи, як SPHINCS+, пропонують найвищий рівень безпеки, базуючись лише на фундаментальних криптографічних примітивах. Хоча вони мають більші розміри підписів, їхня безпека ґрунтується на мінімальних криптографічних припущеннях.

Мультиваріативні підписи представляють третій напрямок розвитку. Незважаючи на те, що вони не були обрані як основні стандарти NIST, дослідження в цій області продовжуються, особливо в контексті специфічних застосувань, де їхні унікальні властивості можуть бути особливо корисними.

Особлива увага приділяється розробці гібридних систем, які дозволяють поступово переходити до постквантових алгоритмів без втрати сумісності з існуючими системами. Цей підхід особливо важливий для критичної інфраструктури, де неможливо миттєво замінити всі криптографічні компоненти.

Порівняння підходів показаний на рисунку 1.4.

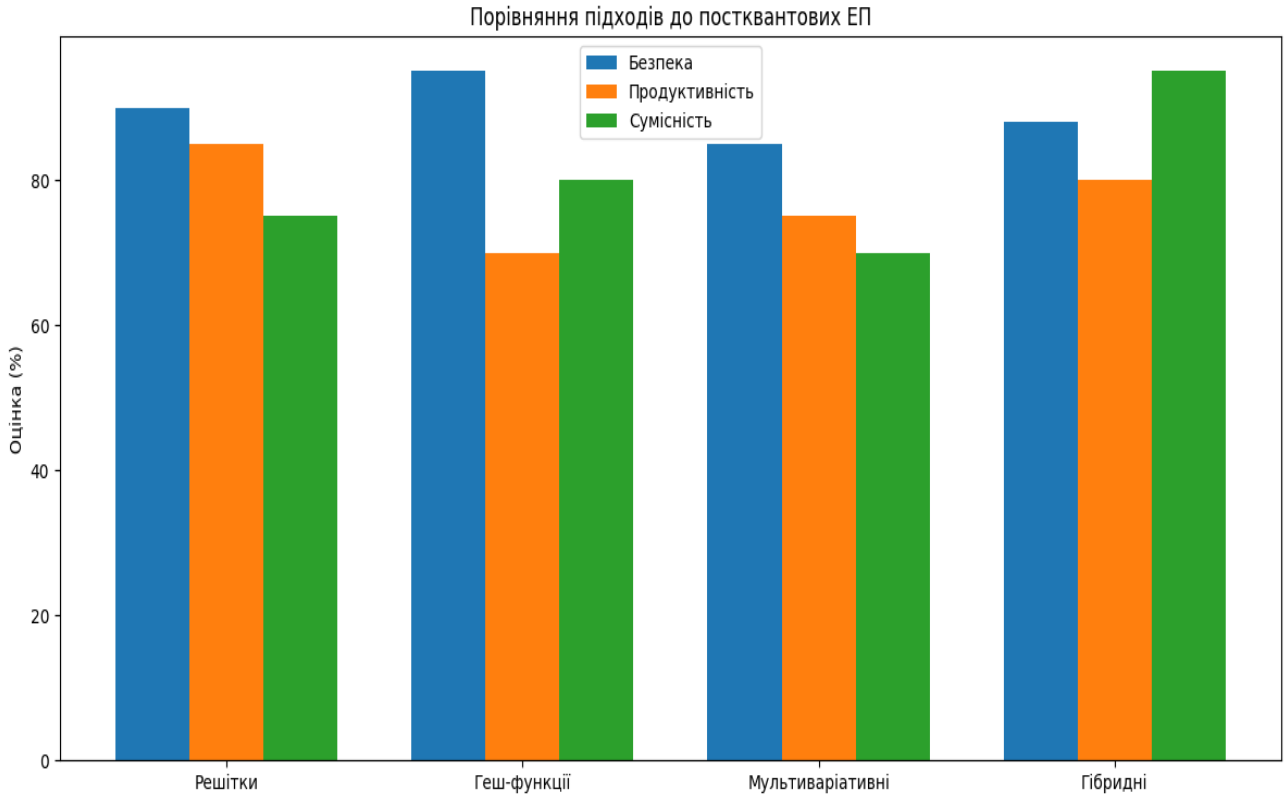

Рисунок 1.4 – Порівняння підходів до постквантових ЕП

Розвиток постквантових ЕП продовжує активно прогресувати, причому азійські розробки, включаючи KAZ-SIGN, вносять значний вклад у цей процес. Різноманітність підходів та активна міжнародна співпраця створюють міцну основу для забезпечення криптографічної безпеки в постквантову еру.

1.2 Огляд вимог NIST США до постквантових криптографічних примітивів

1.2.1 Базові вимоги до безпеки постквантових криптографічних примітивів

Національний інститут стандартів і технологій США (NIST) встановив комплексний набір вимог до безпеки постквантових криптографічних

примітивів. Ці вимоги формують фундаментальну основу для розробки та оцінки нових алгоритмів електронного підпису.

Стійкість до квантових атак є першочерговою вимогою для постквантових криптографічних примітивів. NIST вимагає, щоб алгоритми зберігали свою безпеку навіть при наявності потужного квантового комп'ютера.

Це означає, що алгоритм повинен бути стійким не лише до класичних атак, але й до квантових алгоритмів, таких як алгоритм

Шора та алгоритм Гровера. При цьому особлива увага приділяється оцінці складності квантових атак та необхідним ресурсам для їх реалізації.

NIST визначає п'ять рівнів безпеки для постквантових алгоритмів, кожен з яких відповідає певному рівню захисту від квантових атак. Рівень I забезпечує базовий захист, еквівалентний AES-128 у класичному середовищі.

Рівень II підвищує вимоги до безпеки, наближаючись до рівня SHA-256. Рівень III відповідає AES-192 та забезпечує високий рівень захисту. Рівень V, найвищий, вимагає екстремально високого рівня безпеки, еквівалентного AES-256.

Вимоги до криптографічної стійкості включають стійкість до різних типів криптоаналітичних атак.

Алгоритми повинні забезпечувати повну стійкість до підробки підпису, стійкість до колізій та стійкість до знаходження прообразу.

NIST також вимагає, щоб безпека алгоритму базувалася на добре вивчених математичних проблемах, складність яких може бути точно оцінена як для класичних, так і для квантових обчислень.

Наприклад порівняння рівнів безпеки показані на рисунку 1.5.

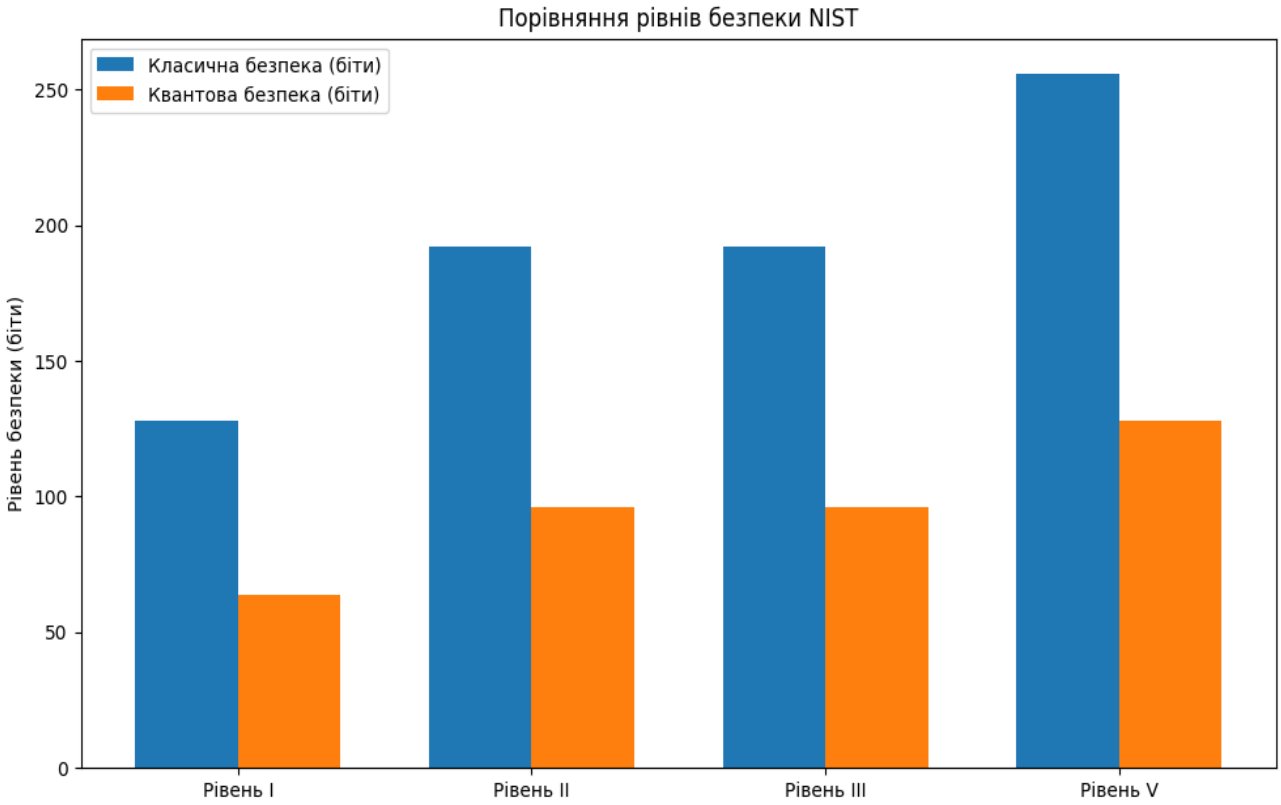

Рисунок 1.5 – Порівняння рівнів безпеки NIST

Важливим аспектом вимог NIST є необхідність доведення безпеки. Кожен алгоритм повинен супроводжуватися формальним доведенням безпеки, яке показує, що зламування алгоритму щонайменше так само складне, як розв'язання певної математичної проблеми, яка вважається складною навіть для квантового комп'ютера.

Окрім цього, NIST вимагає, щоб алгоритми демонстрували стійкість до побічних каналів атак. Це означає, що реалізація алгоритму повинна бути захищена від витоку інформації через час виконання, споживання енергії або електромагнітне випромінювання. Ця вимога особливо важлива для практичного застосування алгоритмів у реальних системах.

Всі ці вимоги формують комплексний підхід до забезпечення безпеки постквантових криптографічних примітивів. Вони враховують не лише

теоретичні аспекти безпеки, але й практичні аспекти реалізації та експлуатації алгоритмів у реальних умовах.

1.2.2 Вимоги до продуктивності постквантових криптографічних примітивів

NIST встановлює суворі вимоги до продуктивності постквантових алгоритмів електронного підпису, враховуючи їх практичне застосування в реальних системах. Ці вимоги охоплюють кілька ключових аспектів, які безпосередньо впливають на ефективність роботи криптосистеми.

Розмір ключів є критичним параметром для постквантових алгоритмів. На відміну від класичних криптосистем, де розміри ключів зазвичай не перевищують кількох кілобайт, постквантові алгоритми часто вимагають значно більших ключів. NIST рекомендує, щоб розмір відкритого ключа не перевищував 32 кілобайти для практичного використання. Це обмеження особливо важливе для вбудованих систем та мобільних пристроїв, де обсяг пам'яті обмежений.

Розмір підпису також є важливим показником ефективності алгоритму. Великі розміри підписів можуть створювати значне навантаження на мережу та системи зберігання даних. NIST встановлює рекомендований максимальний розмір підпису в межах 64 кілобайт, хоча перевага надається алгоритмам з меншими розмірами підписів.

Швидкість генерації ключів повинна бути достатньою для практичного застосування. NIST вимагає, щоб процес генерації пари ключів займав не більше кількох секунд на стандартному обладнанні. Це особливо важливо в системах, де потрібна часта зміна ключів або створення тимчасових ключових пар.

Швидкість формування підпису є критичним параметром для багатьох застосувань, особливо в системах реального часу. Алгоритм повинен забезпечувати формування підпису за час, що не перевищує кількох мілісекунд на сучасних процесорах. Це дозволяє використовувати алгоритм у високонавантажених системах та інтерактивних додатках.

Швидкість перевірки підпису також повинна бути оптимальною, оскільки операція перевірки часто виконується частіше, ніж формування підпису. NIST рекомендує, щоб час перевірки був порівнянним або меншим за час формування підпису.

Наприклад на рисунку 1.6 показано порівняння характеристик постквантових ЕП.

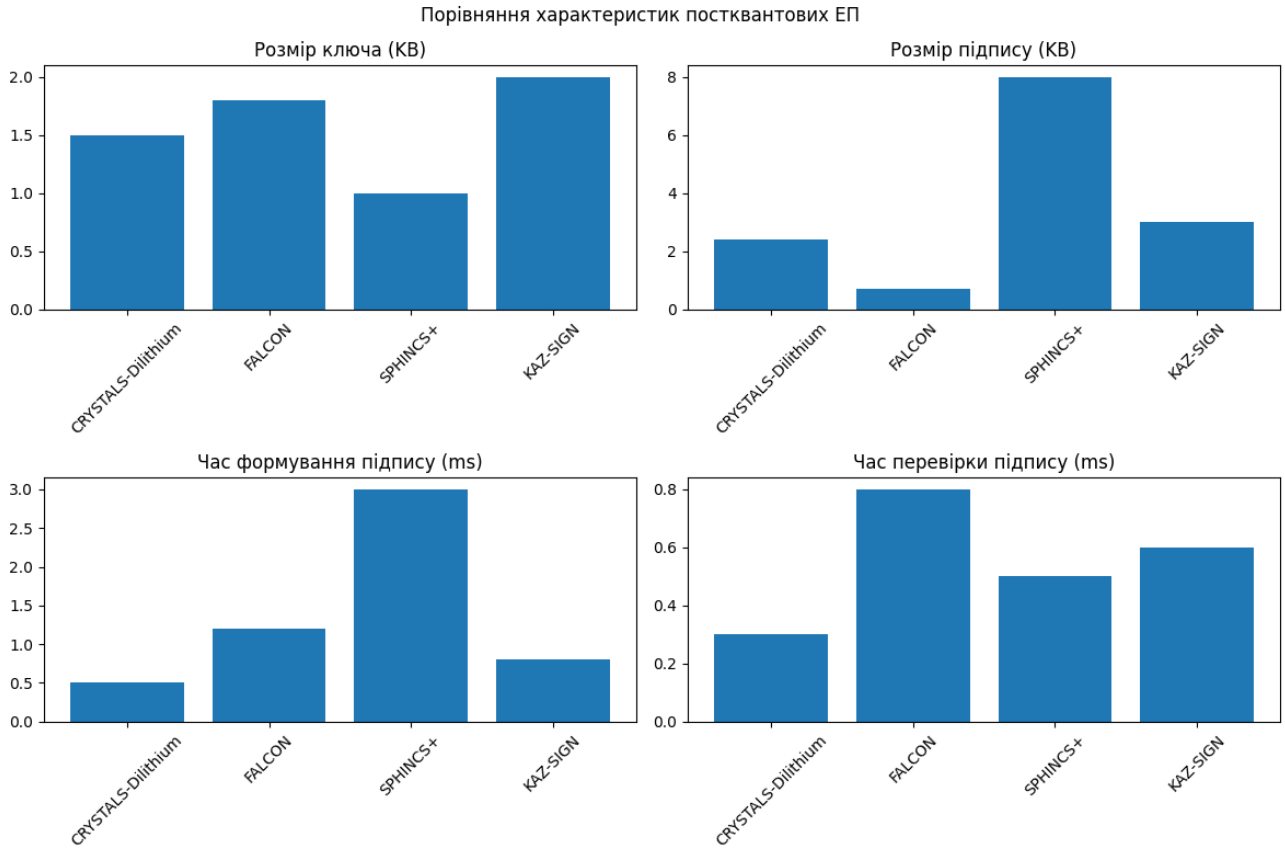

Рисунок 1.6 – Порівняння постквантових ЕП

Важливо зазначити, що всі ці характеристики повинні розглядатися у комплексі, оскільки вони взаємопов'язані. Наприклад, алгоритми з меншими розмірами ключів можуть вимагати більше часу на формування підпису, і навпаки. NIST враховує цей баланс при оцінці алгоритмів, надаючи перевагу рішенням, які демонструють оптимальне співвідношення всіх характеристик.

Ці вимоги до продуктивності є важливим фактором при виборі алгоритмів для стандартизації, оскільки вони безпосередньо впливають на практичну застосовність постквантових криптосистем у реальних умовах.

1.2.3 Додаткові вимоги до постквантових криптографічних примітивів

NIST приділяє особливу увагу додатковим вимогам до постквантових алгоритмів, які забезпечують їх практичну застосовність та безпеку в реальних умовах експлуатації. Ці вимоги доповнюють основні критерії безпеки та продуктивності.

Гнучкість реалізації є важливою характеристикою сучасних криптографічних алгоритмів. Алгоритм повинен ефективно працювати на різних платформах – від потужних серверів до мобільних пристроїв та вбудованих систем. NIST вимагає, щоб алгоритми підтримували різні режими роботи та дозволяли налаштування параметрів відповідно до конкретних потреб застосування. Це включає можливість вибору рівня безпеки, оптимізацію під конкретні апаратні платформи та адаптацію до різних протоколів зв'язку.

Можливість апаратної реалізації стає все більш важливою вимогою, особливо для високопродуктивних систем та пристроїв з обмеженими ресурсами. Алгоритми повинні бути придатними для ефективної реалізації як у програмному, так і в апаратному вигляді. NIST оцінює можливість реалізації на різних апаратних платформах, включаючи FPGA та ASIC. Важливими аспектами є:

- Паралелізація обчислень;
- Оптимізація використання пам'яті;
- Енергоефективність;
- Можливість використання спеціалізованих інструкцій процесора.

Стійкість до побічних атак є критичним аспектом безпеки в реальних умовах експлуатації. Алгоритми повинні бути захищені від атак, що

використовують витік інформації через побічні канали. NIST вимагає, щоб реалізації були стійкими до:

- Часових атак;
- Атак на основі аналізу споживання енергії;
- Електромагнітних атак;
- Атак на основі аналізу помилок.

Наприклад на рисунку 1.7 показана вразливість до побічних атак при різних рівнях захисту.

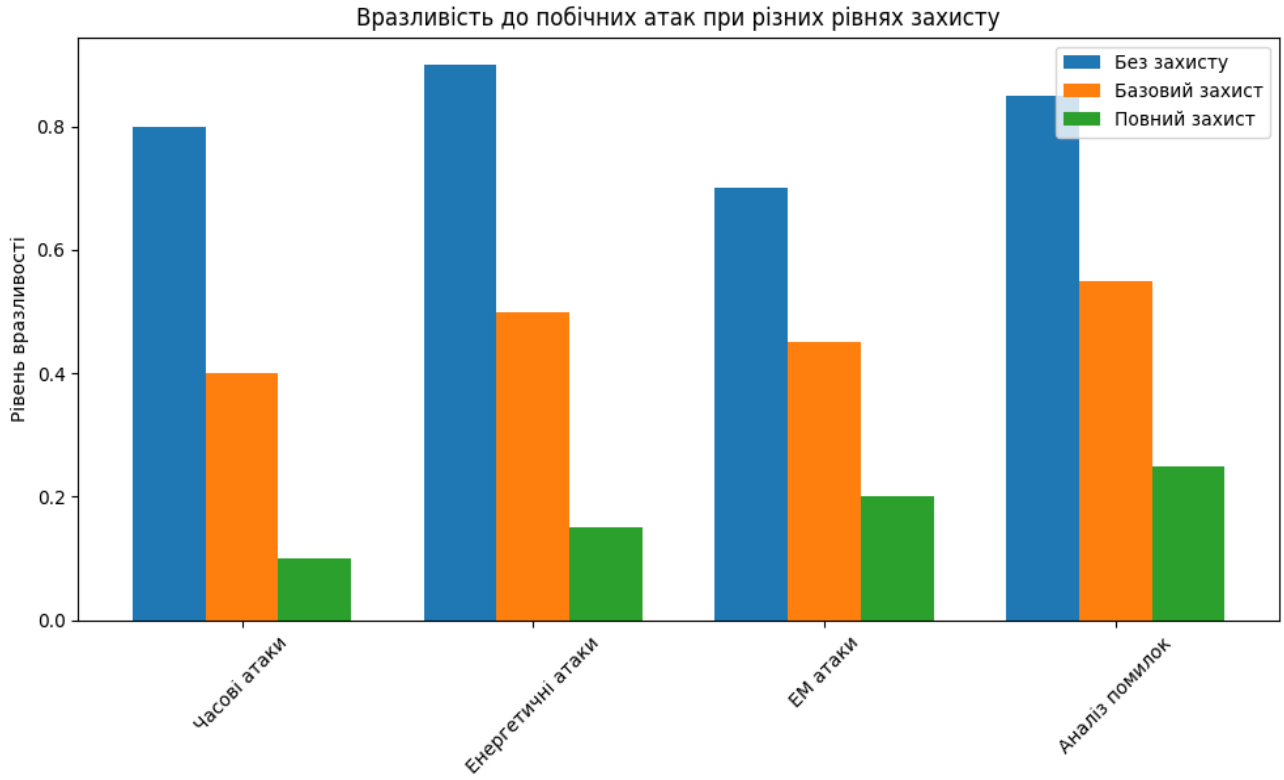

Рисунок 1.7 – Вразливість до побічних атак при різних рівнях захисту

Практична реалізація цих вимог вимагає комплексного підходу до розробки та впровадження постквантових алгоритмів. Розробники повинні враховувати не лише математичні аспекти безпеки, але й особливості реальної експлуатації систем. Важливим є забезпечення балансу між рівнем захисту від побічних атак та продуктивністю системи.

Додаткові вимоги NIST також включають необхідність детальної документації та відкритості реалізації. Це дозволяє проводити незалежний аудит безпеки та сприяє широкому впровадженню алгоритмів. Особлива увага приділяється наявності тестових векторів та інструментів верифікації, що полегшують процес впровадження та перевірки коректності реалізації.

Ці додаткові вимоги формують важливу частину загальної оцінки постквантових алгоритмів та їх придатності для практичного використання. Вони забезпечують не лише теоретичну безпеку, але й практичну застосовність алгоритмів у реальних умовах експлуатації.

1.3 Порівняльний аналіз існуючих кандидатів на постквантові ЕП

1.3.1 Класифікація постквантових електронних підписів

Сучасні постквантові електронні підписи можна розділити на кілька основних категорій, кожна з яких базується на різних математичних принципах та має свої унікальні характеристики.

Підписи на основі решіток.

CRYSTALS-Dilithium та FALCON представляють найбільш перспективний напрямок розвитку постквантових ЕП. Ці алгоритми базуються на складності задач у решітках, зокрема на проблемі навчання з помилками (LWE) та її варіаціях. Dilithium використовує модульні решітки та оптимізований для ефективної реалізації підхід, що забезпечує хороший баланс між розміром підпису та швидкодією. FALCON, у свою чергу, використовує NTRU решітки та швидкі перетворення Фур'є для досягнення менших розмірів підпису, хоча це призводить до складнішої реалізації [5-7].

Підписи на основі геш-функцій.

SPHINCS+ представляє принципово інший підхід, базуючись виключно на властивостях криптографічних геш-функцій. Цей алгоритм використовує структуру дерева Меркла та одноразові підписи для створення статистично безпечної схеми підпису. Головною перевагою SPHINCS+ є мінімальні

криптографічні припущення щодо безпеки, оскільки він спирається лише на стійкість геш-функцій. Проте це призводить до більших розмірів підпису та нижчої швидкодії порівняно з рішеннями на основі решіток.

Підписи на основі мультиваріативних перетворень.

Мультиваріативна криптографія базується на складності розв'язання систем поліноміальних рівнянь багатьох змінних. Хоча жоден з мультиваріативних алгоритмів не був обраний як фіналіст конкурсу NIST, цей напрямок залишається активною областю досліджень.

Основною перевагою таких систем є висока швидкість верифікації підпису, проте вони часто страждають від великих розмірів ключів.

Інші підходи.

Існують також альтернативні підходи до побудови постквантових ЕП, включаючи системи на основі кодів, виправляючих помилки, та ізогеній суперсингулярних еліптичних кривих.

Хоча ці підходи мають цікаві теоретичні властивості, вони поки що не досягли такого рівня практичної застосовності, як рішення на основі решіток або геш-функцій.

Кожен з цих підходів має свої переваги та недоліки, що робить їх придатними для різних сценаріїв використання.

Рішення на основі решіток наразі вважаються найбільш перспективними завдяки хорошому балансу характеристик.

Підписи на основі геш-функцій забезпечують найвищий рівень довіри до безпеки, але за рахунок продуктивності.

Мультиваріативні системи та інші підходи можуть знайти своє застосування в специфічних сценаріях, де їхні унікальні властивості є особливо важливими.

Наприклад на рисунку 1.8 показана порівняння різних підходів.

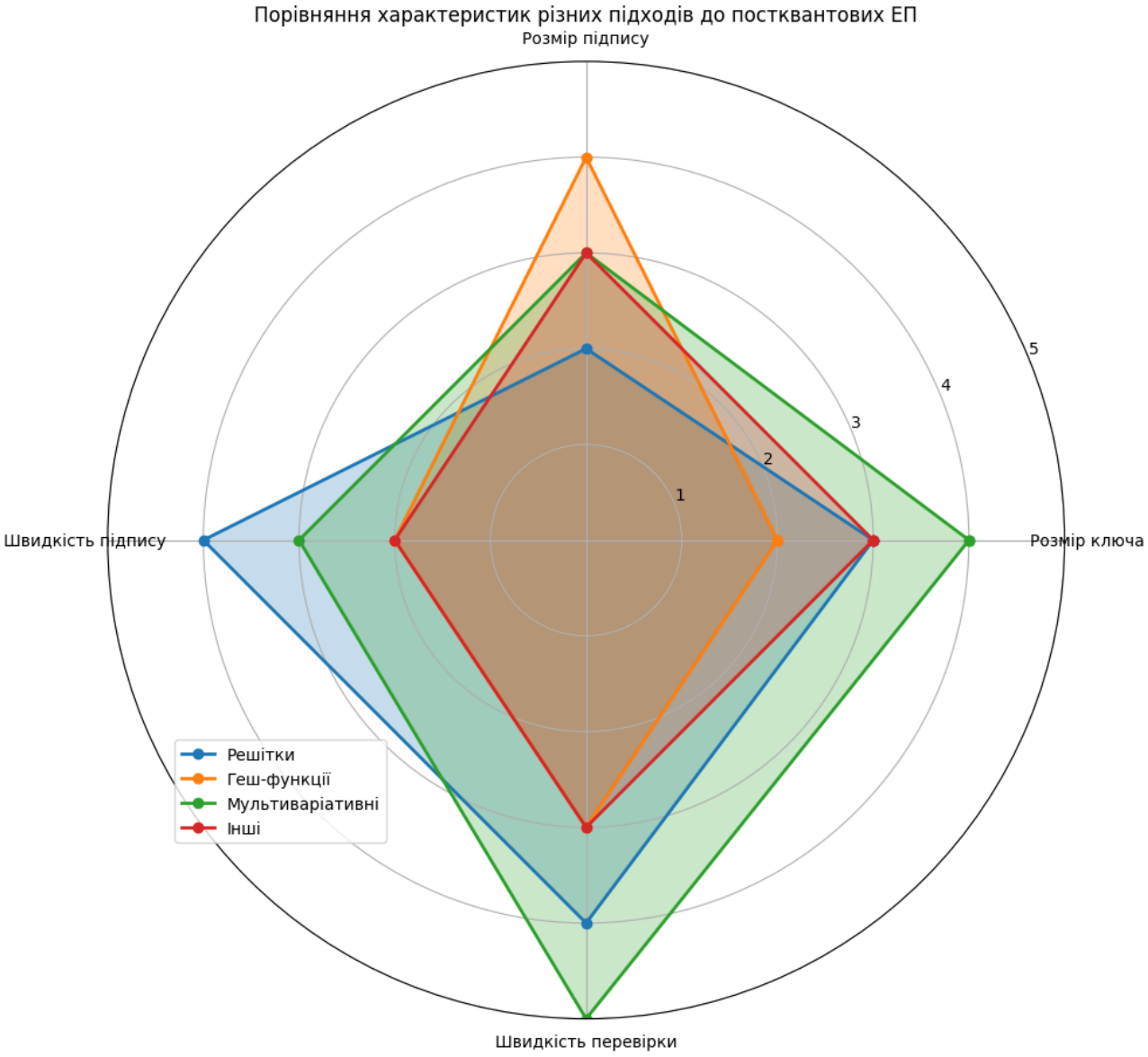

Рисунок 1.8 – Порівняння характеристик різних підходів до постквантових ЕП

Розуміння цієї класифікації та характеристик різних підходів є критично важливим для вибору відповідного алгоритму для конкретного застосування та подальшого розвитку постквантової криптографії в цілому.

1.3.2 Порівняльний аналіз постквантових ЕП за основними критеріями

Порівняльний аналіз постквантових електронних підписів за ключовими критеріями дозволяє оцінити їх практичну застосовність та визначити оптимальні сфери використання кожного підходу.

Криптографічна стійкість.

Криптографічна стійкість є фундаментальним критерієм оцінки алгоритмів ЕП. SPHINCS+ демонструє найвищий рівень теоретичної безпеки, оскільки базується лише на властивостях криптографічних геш-функцій. Dilithium та FALCON забезпечують безпеку на основі добре вивчених проблем теорії решіток, хоча їх стійкість залежить від складності специфічних математичних задач. Мультиваріативні підписи, хоча й пропонують потенційно високу стійкість, мають менш досліджену теоретичну базу [7].

Розміри ключів та підписів.

Розміри криптографічних параметрів суттєво відрізняються між різними підходами. FALCON демонструє найкращі показники щодо розміру підпису (близько 1 КБ), але має більші ключі. Dilithium пропонує збалансоване рішення з помірними розмірами як ключів, так і підписів. SPHINCS+ характеризується значно більшими розмірами підписів (понад 8 КБ), що може обмежувати його застосування в деяких сценаріях. Мультиваріативні схеми зазвичай мають дуже великі відкриті ключі.

Обчислювальна складність.

Алгоритми на основі решіток (Dilithium, FALCON) демонструють найкращу продуктивність у більшості операцій. SPHINCS+ має значно вищу обчислювальну складність при формуванні підпису, але помірну при верифікації. Мультиваріативні підписи часто показують високу швидкість верифікації, але можуть бути повільними при генерації підпису. Наприклад на рисунку 1.9 показано порівняльний аналіз.

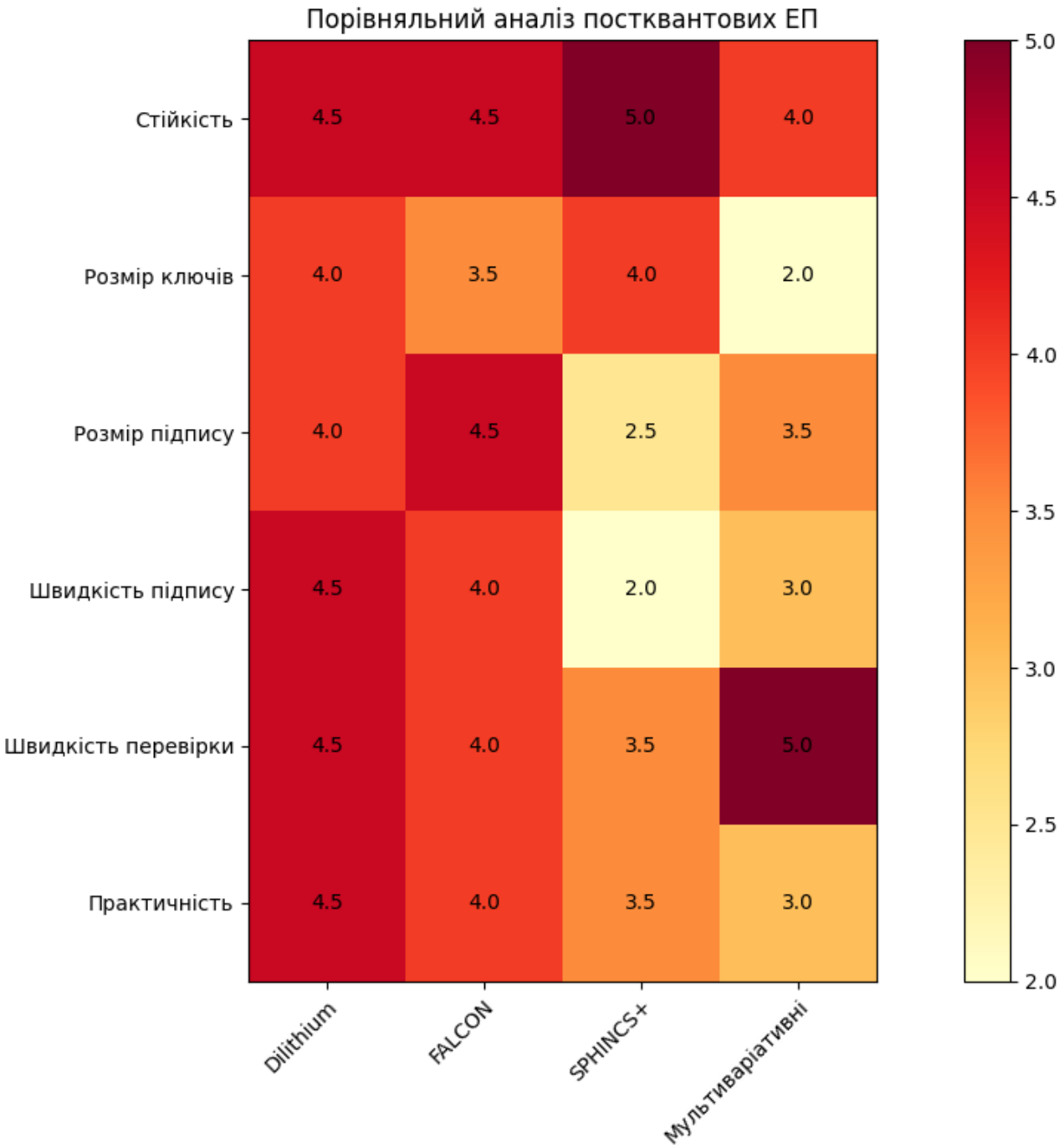

Рисунок 1.9 – Порівняльний аналіз постквантових ЕП

Практична реалізованість.

Практична реалізованість враховує такі аспекти як:

- Простота реалізації;
- Можливість оптимізації;

- Стійкість до побічних атак;
- Гнучкість налаштування параметрів.

Dilithium демонструє найкращу практичну реалізованість завдяки простій структурі та ефективним оптимізаціям. FALCON, хоча й забезпечує кращі розміри підпису, вимагає складнішої реалізації через використання операцій з числами з плаваючою комою. SPHINCS+ має відносно просту реалізацію, але обмежену практичність через великі розміри підписів. Мультиваріативні схеми часто стикаються з викликами при практичній реалізації через складність основних операцій [8-9].

Загальний аналіз показує, що алгоритми на основі решіток, особливо Dilithium, наразі пропонують найкращий компроміс між різними характеристиками. Це пояснює їх вибір як основних кандидатів у стандартизації NIST. Проте інші підходи, зокрема SPHINCS+, залишаються важливими альтернативами, особливо в контекстах, де пріоритетом є максимальна теоретична безпека.

Вибір конкретного алгоритму повинен базуватися на специфічних вимогах застосування, враховуючи баланс між безпекою, продуктивністю та практичними обмеженнями реалізації.

1.3.3 Детальний аналіз алгоритму KAZ-SIGN

Математична основа.

KAZ-SIGN базується на математичному апараті алгебраїчних решіток, але має свої унікальні особливості в порівнянні з іншими решітковими підходами. Алгоритм використовує спеціальну структуру локальних кілець та модифіковану версію проблеми навчання з помилками (LWE).

Основні математичні компоненти включають:

- Кільце поліномів:

$$R = Z[x]/(x^n + 1) \quad (1.1)$$

Де, x - це змінна полінома;

n - це степінь полінома (зазвичай вибирається як степінь двійки, наприклад 256, 512 або 1024);

(x^n + 1) - це поліном, за модулем якого відбуваються обчислення.

- Модульне відображення по q;
- Спеціальні розподіли для генерації помилок;
- Модифіковану схему відбору векторів.

Особливості реалізації.

Алгоритм KAZ-SIGN має трирівневу структуру реалізації.

1) Генерація ключів:

- Створення секретного поліному;
- Обчислення відкритого ключа;
- Формування додаткових параметрів.

2) Формування підпису:

- Генерація випадкових значень;
- Обчислення зобов'язання;
- Формування відповіді на виклик.

3) Верифікація:

- Перевірка формату підпису;
- Відновлення зобов'язання;
- Валідація відповіді.

На рисунку 1.10 показано порівняння продуктивності операцій.

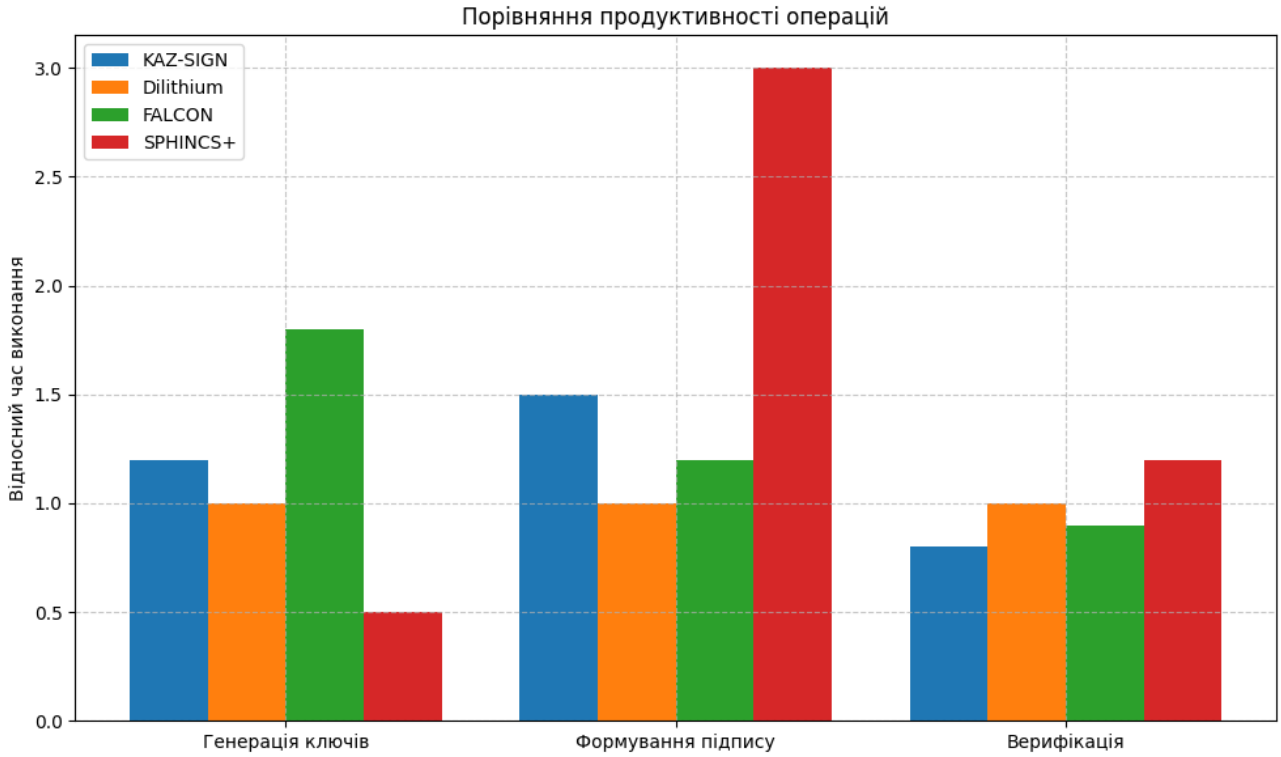

Рисунок 1.10 – Порівняння продуктивності операцій

Переваги.

1) Оптимізована структура:

- Ефективне використання пам'яті;
- Швидка верифікація підпису;
- Можливість апаратної оптимізації.

2) Безпека:

- Базується на добре вивчених математичних проблемах;
- Стійкість до відомих квантових атак;
- Додаткові механізми захисту від побічних каналів.

3) Практичність:

- Гнучкі параметри безпеки;
- Сумісність з існуючими протоколами;
- Можливість вбудованої реалізації.

Недоліки.

1. Технічні обмеження:

- Відносно великі розміри ключів;
- Складність реалізації деяких операцій;
- Вимоги до якості генератора випадкових чисел.

2. Практичні аспекти:

- Обмежена кількість незалежних реалізацій;
- Потреба в додатковій валідації безпеки;
- Складність інтеграції в деякі існуючі системи.

Характеристики KAZ-SIGN показана на рисунку 1.11.

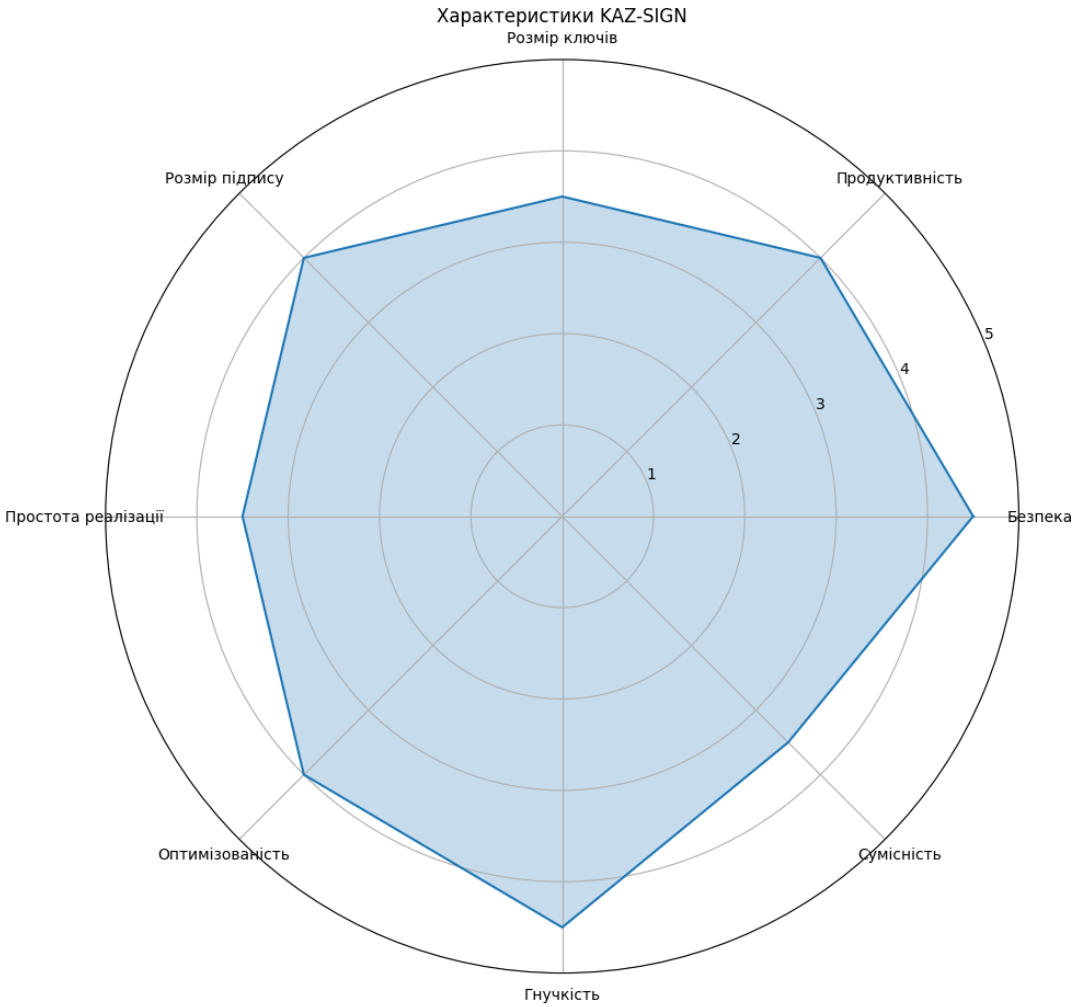

Рисунок 1.11 - Характеристики KAZ-SIGN

KAZ-SIGN представляє собою цікавий підхід до постквантової криптографії, пропонуючи деякі унікальні оптимізації та рішення. Хоча алгоритм має певні обмеження, його загальна архітектура та характеристики роблять його перспективним кандидатом для специфічних застосувань, особливо в системах з обмеженими ресурсами.

Подальший розвиток алгоритму може включати:

- Оптимізацію розмірів ключів;
- Покращення продуктивності критичних операцій;
- Розширення можливостей практичного застосування;
- Додаткову валідацію безпеки.

1.4 Висновки до розділу

У першому розділі було проведено комплексний аналіз сучасного стану постквантової криптографії та електронних підписів, що дозволяє зробити наступні ключові висновки.

Проведений аналіз міжнародних стандартів та вимог NIST показав формування чіткої системи критеріїв для постквантових криптографічних примітивів. Особливо важливим є встановлення балансу між безпекою, продуктивністю та практичною реалізованістю алгоритмів. Стандартизація NIST створює надійну основу для подальшого розвитку галузі.

Дослідження показало наявність декількох фундаментально різних підходів до побудови постквантових ЕП. Решіткові підписи демонструють оптимальний баланс характеристик, геш-основані підписи забезпечують найвищий рівень теоретичної безпеки, а мультиваріативні підписи пропонують цікаві альтернативні властивості.

Детальне порівняння існуючих рішень виявило, що CRYSTALS-Dilithium та FALCON є найбільш збалансованими рішеннями, SPHINCS+ забезпечує

найвищий рівень довіри до безпеки, а KAZ-SIGN пропонує цікаві оптимізації для специфічних застосувань.

Аналіз сучасних тенденцій вказує на продовження досліджень у напрямку оптимізації існуючих алгоритмів, важливість розробки гібридних рішень та необхідність покращення практичної реалізованості постквантових алгоритмів.

На основі проведеного аналізу можна сформулювати практичні рекомендації щодо вибору та впровадження постквантових алгоритмів. При виборі алгоритму необхідно враховувати специфічні вимоги конкретного застосування. Доцільно розглядати можливість використання гібридних рішень для забезпечення плавного переходу. Особливу увагу слід приділяти питанням практичної реалізації та захисту від побічних атак. Важливо забезпечити можливість оновлення параметрів безпеки в майбутньому.

Проведений аналіз дозволив визначити перспективні напрямки подальших досліджень, включаючи оптимізацію існуючих алгоритмів для специфічних платформ та застосувань, розробку нових методів захисту від побічних атак, дослідження можливостей комбінування різних підходів та вдосконалення методів оцінки криптографічної стійкості.

Важливим аспектом, виявленим під час аналізу, є необхідність врахування специфіки різних обчислювальних платформ при розробці та впровадженні постквантових рішень. Особливо це стосується мобільних та вбудованих систем, де ресурсні обмеження накладають додаткові вимоги на реалізацію криптографічних алгоритмів.

Окремої уваги заслуговує питання сумісності нових постквантових алгоритмів з існуючими криптографічними протоколами та інфраструктурою. Аналіз показав необхідність розробки ефективних механізмів міграції та забезпечення зворотної сумісності при впровадженні нових рішень.

Суттєвим викликом залишається питання стандартизації методів тестування та оцінки постквантових алгоритмів. Необхідно розробити

уніфіковані підходи до оцінки продуктивності та безпеки, які враховуватимуть як класичні, так і квантові загрози, а також специфіку різних сценаріїв використання.

Таким чином, проведене дослідження формує комплексне розуміння сучасного стану постквантової криптографії та створює основу для подальшого розвитку цієї галузі. Отримані результати можуть бути використані при розробці та впровадженні постквантових криптографічних рішень в реальних системах.

2 ТЕОРЕТИЧНЕ ОБҐРУНТУВАННЯ ТА ДОСЛІДЖЕННЯ МЕТОДУ KAZ-SIGN

2.1 Математичний апарат та криптографічні примітиви KAZ-SIGN

Математичний апарат KAZ-SIGN базується на кільці поліномів. Всі обчислення виконуються за модулем простого числа q, що формує кільце:

$$Rq = (Z/qZ)[x]/(x^n + 1) \quad (2.1)$$

Вибір таких параметрів забезпечує ефективність операцій та необхідний рівень криптографічної стійкості.

Важливим компонентом схеми є використання спеціальних розподілів ймовірностей для генерації випадкових поліномів. KAZ-SIGN використовує дискретний гаусівський розподіл з параметром σ для генерації секретних ключів та шуму. Цей розподіл забезпечує необхідний баланс між безпекою та ефективністю, оскільки генеровані значення мають обмежену норму, що важливо для коректної роботи схеми.

Для векторних просторів над кільцем Rq визначаються спеціальні норми, що враховують поліноміальну природу елементів. Використовується l2-норма для векторів та евклідова норма для поліномів. Ці метрики відіграють ключову роль у визначенні умов коректності підпису та оцінці складності атак.

Серед базових криптографічних примітивів KAZ-SIGN використовує геш-функцію H для обчислення геш-значень повідомлень та генерації випадкових чисел. Вибір конкретної геш-функції (наприклад, SHA-3) визначається вимогами до безпеки та продуктивності. Генератор псевдовипадкових чисел (PRNG) використовується для отримання випадкових значень при генерації ключів та формуванні підпису [10-12].

Особливістю KAZ-SIGN є модифікована версія проблеми LWE (Learning With Errors), адаптована для роботи в кільці поліномів. Ця модифікація дозволяє

зменшити розміри ключів та підвищити ефективність обчислень при збереженні необхідного рівня безпеки. Складність цієї проблеми базується на складності знаходження коротких векторів у спеціальних решітках над кільцем поліномів.

Для забезпечення детермінованості підпису використовується механізм детермінованої генерації випадкових значень на основі секретного ключа та повідомлення. Це дозволяє уникнути проблем з якістю випадкових чисел та спрощує реалізацію схеми на практиці.

Всі ці математичні компоненти разом формують цілісну криптографічну систему, що забезпечує необхідний рівень безпеки в постквантовому середовищі при збереженні прийнятної продуктивності та практичної реалізованості.

2.1.1 Математичні проблеми та модифікована версія LWE

Безпека алгоритму KAZ-SIGN базується на складності вирішення фундаментальних математичних проблем у кільці поліномів. Основною з них є проблема пошуку найкоротшого вектора (SVP - Shortest Vector Problem) у спеціальних решітках над кільцем:

$$R = Z[x]/(x^n + 1) \quad (2.2)$$

Ця проблема вважається обчислювально складною навіть для квантових комп'ютерів, що забезпечує постквантову стійкість схеми.

Модифікована версія проблеми LWE, що використовується в KAZ-SIGN, формулюється наступним чином. Нехай $a \in Rq^k$ - випадковий вектор, $s \in Rq^k$ - секретний вектор з коефіцієнтами малої норми, e - вектор помилок, згенерований за дискретним гаусівським розподілом. Тоді LWE-зразок має вигляд:

$$(a, b = \langle a,s \rangle + e \bmod q) \quad (2.3)$$

Де $\langle a,s \rangle$ позначає скалярний добуток векторів у кільці Rq.

Модифікація класичної LWE проблеми полягає у використанні структурованих матриць та спеціальних розподілів для генерації векторів. Це дозволяє значно зменшити розмір відкритого ключа при збереженні необхідного

рівня безпеки. Складність знаходження s за набором LWE-зразків еквівалентна вирішенню певних варіантів проблеми SVP у відповідних решітках.

Додатковою математичною проблемою, що забезпечує безпеку KAZ-SIGN, є проблема знаходження малої лінійної комбінації (SIS - Short Integer Solution) у тому ж кільці. Ця проблема використовується при верифікації підпису та тісно пов'язана з проблемою SVP.

Важливою особливістю модифікованої версії LWE є використання спеціальної структури кільця поліномів, що дозволяє ефективно виконувати множення елементів за допомогою швидкого перетворення Фур'є (NTT). Це суттєво покращує продуктивність схеми порівняно з класичною версією LWE.

Теоретична безпека схеми базується на доведенні того, що успішна атака на KAZ-SIGN призводить до ефективного алгоритму вирішення модифікованої LWE проблеми, що вважається обчислювально складним завданням. Рівень безпеки визначається вибором параметрів n, q та σ, які повинні забезпечувати достатню складність найкращих відомих алгоритмів для вирішення відповідних математичних проблем.

Для забезпечення практичної безпеки важливо також враховувати можливість гібридних атак, що комбінують класичні та квантові алгоритми. Тому параметри схеми вибираються з урахуванням складності як класичних, так і квантових алгоритмів решітчастої редукції.

Таким чином, математична основа KAZ-SIGN забезпечує необхідний рівень постквантової безпеки при збереженні ефективності практичної реалізації завдяки використанню спеціальних алгебраїчних структур та оптимізованих алгоритмів.

2.2 Алгоритми генерування загальних параметрів та ключів

Процес генерування параметрів та ключів у системі KAZ-SIGN є фундаментальним етапом, що визначає загальну безпеку та ефективність схеми. Розглянемо детально процес вибору та генерації основних параметрів системи.

Вибір системних параметрів починається з визначення степеня n полінома, який повинен бути степенем двійки для забезпечення ефективності операцій в кільці.

Для забезпечення 128-бітного рівня безпеки рекомендується використовувати n = 1024 [13-14]. Модуль q вибирається як просте число, що задовольняє умові q ≡ 1 (mod 2n), що необхідно для ефективного використання NTT перетворення. Типове значення q знаходиться в діапазоні 2^23 - 2^30.

Параметри розподілів ймовірностей вибираються таким чином, щоб забезпечити необхідний баланс між безпекою та ефективністю. Для дискретного гаусівського розподілу параметр σ визначається як:

$$\sigma = \alpha \cdot q/\sqrt{(2\pi)} \quad (2.4)$$

Де α - допоміжний параметр, що впливає на рівень безпеки.

Процедура генерації загальних параметрів включає наступні кроки:

1) Генерація базового полінома:

$$f(x) = x^n + 1; \quad (2.5)$$

2) Вибір простого модуля q;
3) Ініціалізація генератора псевдовипадкових чисел;
4) Формування параметрів розподілів.

Важливим аспектом є перевірка коректності вибраних параметрів. Для цього проводиться аналіз:

- Складності найкращих відомих атак;
- Ймовірності помилок декодування;
- Ефективності базових операцій.

Всі параметри повинні забезпечувати:

- Достатній рівень криптографічної стійкості;
- Прийнятну ймовірність коректної роботи схеми;
- Ефективність обчислень на цільових платформах.

Після генерації загальних параметрів вони можуть бути використані для генерації ключових пар. Важливо зазначити, що параметри системи є публічними та можуть бути стандартизовані для конкретних застосувань.

Процес генерації параметрів є критичним для безпеки всієї системи, тому він повинен виконуватися з особливою ретельністю та з використанням якісних джерел ентропії. Згенеровані параметри повинні бути належним чином задокументовані та перевірені перед використанням у реальних системах.

2.2.1 Генерація та зберігання ключів KAZ-SIGN

Алгоритм генерації ключової пари в KAZ-SIGN базується на використанні попередньо згенерованих системних параметрів.

Процес починається з генерації секретного ключа як вектора поліномів з коефіцієнтами, розподіленими за дискретним гаусівським розподілом. На основі секретного ключа обчислюється відкритий ключ шляхом застосування спеціальних поліноміальних перетворень.

Формат зберігання ключів розроблений з урахуванням вимог сумісності та безпеки. Секретний ключ зберігається у захищеному форматі з використанням додаткового шифрування.

Структура включає метадані, що містять інформацію про використані параметри та версію алгоритму. Відкритий ключ зберігається у компактному форматі, оптимізованому для ефективної передачі.

Наприклад алгоритм генерації зображений на рисунку 2.1.

Оцінка складності генерації ключів показує, що основні обчислювальні витрати припадають на операції в кільці поліномів та генерацію випадкових значень за заданим розподілом. Загальна часова складність процесу становить O(n log n) операцій, де n - степінь базового полінома. Просторова складність лінійно залежить від розміру параметра n та становить O(n) слів пам'яті.

Рисунок 2.1 – Алгоритм генерації

Важливим аспектом є забезпечення якості випадкових чисел при генерації ключів. Для цього використовуються криптографічно стійкі генератори псевдовипадкових чисел та, за можливості, апаратні джерела ентропії. Це

критично важливо для безпеки всієї схеми, оскільки недостатня випадковість може призвести до вразливостей.

Процес генерації ключів включає також додаткові перевірки коректності згенерованих значень. Перевіряється відповідність норм векторів заданим обмеженням та виконання необхідних алгебраїчних співвідношень.

Це дозволяє виявити потенційні проблеми на етапі генерації та забезпечити надійну роботу схеми.

Для практичного використання рекомендується періодична зміна ключів, при цьому частота ротації залежить від конкретного застосування та вимог безпеки.

Процедура оновлення ключів повинна виконуватися з дотриманням усіх вимог щодо генерації та безпечного зберігання криптографічних параметрів.

2.3 Алгоритми формування та перевірки електронного підпису

Алгоритм формування електронного підпису в KAZ-SIGN реалізує детерміністичний підхід, що забезпечує унікальність підпису для кожної пари повідомлення-ключ. Процес базується на використанні геш-функцій та спеціальних поліноміальних перетворень у кільці Rq [15-16].

На рисунку 2.2 показаний сам алгоритм.

Процедура перевірки підпису виконує зворотні обчислення для підтвердження коректності підпису.

Верифікатор використовує відкритий ключ для перевірки математичних співвідношень та обмежень на норми векторів.

Формат підпису розроблений для компактного представлення та ефективної передачі. Підпис складається з двох компонентів: вектора z та геш-значення c. Додатково включаються метадані для ідентифікації версії алгоритму та використаних параметрів.

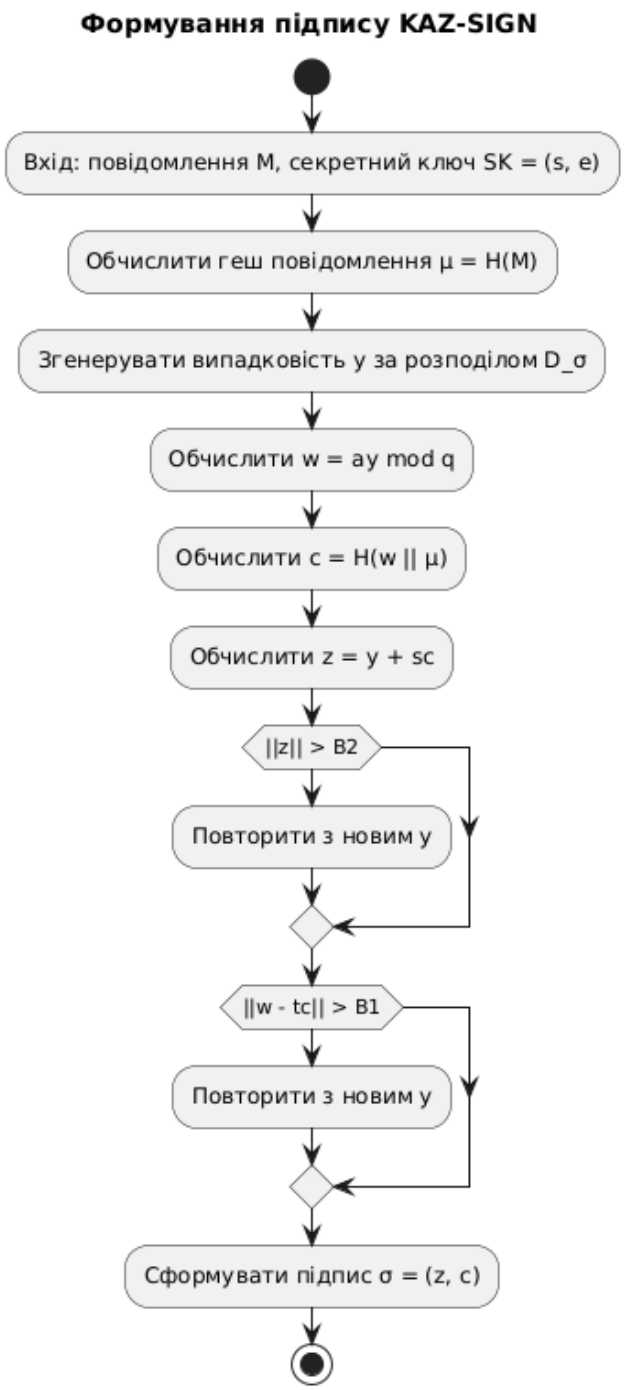

Рисунок 2.2 – Алгоритм формування електронного підпису

Алгоритм перевірки показаний на рисунку 2.3.

Для підвищення продуктивності використовується ряд оптимізацій:

- Використання NTT перетворення для ефективного множення поліномів;

- Попередня обробка відкритого ключа;
- Паралельна обробка незалежних операцій;
- Оптимізація операцій з розрідженими векторами.

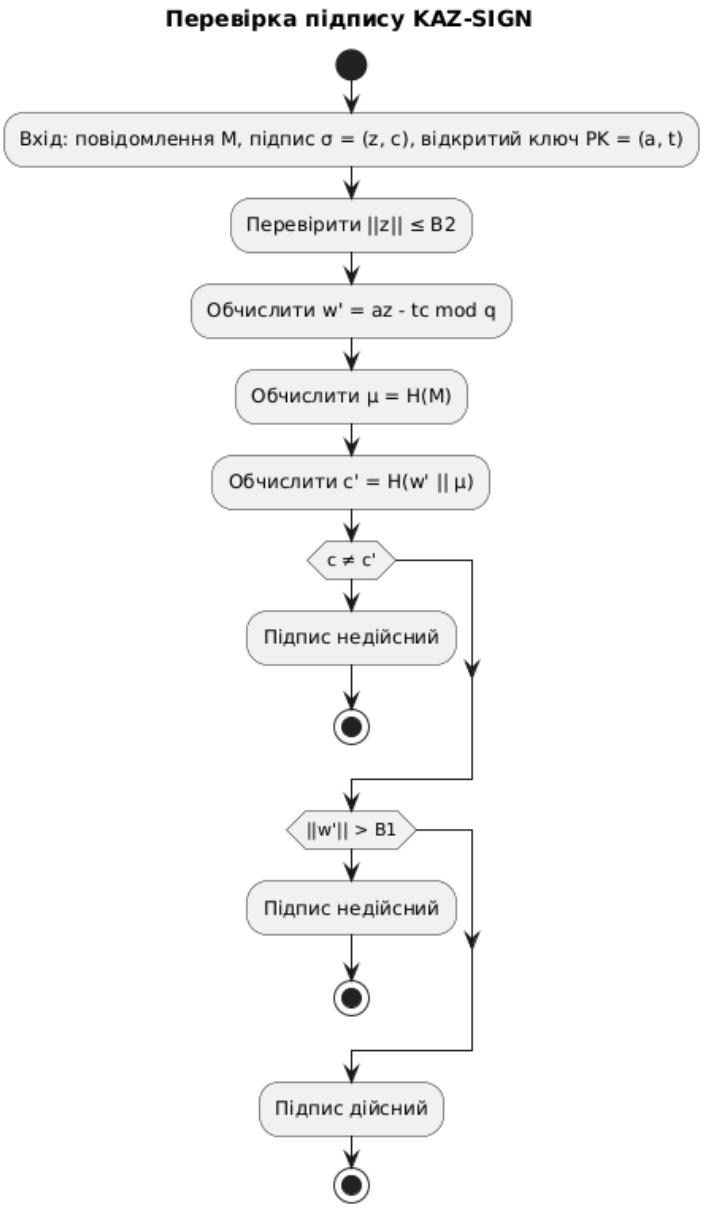

Рисунок 2.3 – Алгоритм перевірки підпису

Оцінка обчислювальної складності показує, що основні витрати припадають на операції множення поліномів та обчислення геш-функцій. Часова складність формування підпису становить O(n log n) операцій, де n - степінь

базового полінома. Процедура перевірки має аналогічну асимптотичну складність, але на практиці виконується швидше завдяки відсутності необхідності генерації випадкових значень [17-21].

Просторова складність алгоритмів лінійно залежить від розміру параметра n. Для формування підпису потрібно O(n) слів пам'яті для зберігання проміжних результатів. Процедура перевірки вимагає меншого об'єму пам'яті, оскільки не потребує зберігання додаткових векторів.

Важливою характеристикою є детермінованість процесу формування підпису, що забезпечує відтворюваність результатів та спрощує тестування реалізацій. При цьому зберігається криптографічна стійкість завдяки використанню геш-функцій та складності базових математичних проблем.

2.4 Аналіз криптографічної стійкості KAZ-SIGN

Теоретична безпека KAZ-SIGN базується на складності вирішення модифікованої проблеми LWE в кільці поліномів. Доведення безпеки схеми використовує редукцію до відомих складних проблем теорії решіток. При цьому важливо, що безпека зберігається навіть при наявності квантового комп'ютера, оскільки найкращі відомі квантові алгоритми не дають суттєвого прискорення у вирішенні базових проблем.

Серед можливих атак на схему можна виділити кілька основних категорій котрі вказані на рисунку 2.4.

Квантові атаки на KAZ-SIGN можуть використовувати алгоритм Шора та його модифікації, проте їх ефективність обмежена через специфіку решіткових проблем. Оцінка складності показує, що для досягнення 128-бітного рівня квантової безпеки достатньо використовувати параметри, що забезпечують приблизно 256-бітний класичний рівень безпеки.

Рисунок 2.4 – Основні категорії атак

Аналіз стійкості до атак побічними каналами виявив потенційні вразливості до:

- Часових атак на операції з поліномами;
- Атак на основі аналізу енергоспоживання;
- Атак на основі помилок обчислень;
- Атак на реалізацію генератора випадкових чисел.

Для протидії цим атакам реалізовано наступні захисні механізми:

- Константний час виконання критичних операцій;
- Маскування проміжних значень;
- Перевірка цілісності обчислень;
- Додаткова валідація параметрів.

Порівняльний аналіз з іншими постквантовими схемами показаний на рисунку 2.5.

Рисунок 2.5 – Порівняння схем ЕП

KAZ-SIGN демонструє збалансовані характеристики безпеки порівняно з іншими схемами. Основними перевагами є:

- Простіша структура порівняно з FALCON;
- Кращі розміри параметрів ніж у SPHINCS+;
- Додаткові оптимізації порівняно з базовим Dilithium.

При цьому схема зберігає необхідний рівень безпеки та має формальні доведення стійкості. Важливою перевагою є можливість гнучкого налаштування параметрів для різних сценаріїв використання при збереженні гарантій безпеки.

Загальний аналіз показує, що KAZ-SIGN забезпечує достатній рівень криптографічної стійкості для практичного використання, включаючи постквантовий сценарій. При цьому необхідно дотримуватись рекомендацій щодо реалізації та вибору параметрів для забезпечення захисту від відомих атак.

2.5 Оцінка алгоритму генерації ключів

При оцінці загальних параметрів KAZ-SIGN необхідно враховувати три ключові аспекти:

Рівень криптографічної стійкості:

- Для забезпечення 128-бітної класичної та 64-бітної квантової безпеки необхідно:
- Степінь поліному $n \geq 512$;
- Модуль $q \geq 12289$;
- Параметр шуму $\sigma \approx 1.0$.

Обчислювальна ефективність:

- Оптимальні значення параметрів для швидкодії:
- Розмір модуля $q < 2^{16}$ для ефективних арифметичних операцій;
- Степінь n повинен бути степенем двійки для швидкого NTT;
- Співвідношення q/σ має бути в межах 8000-15000.

Ресурсні вимоги:

- Оцінка вимог до пам'яті:
- Розмір відкритого ключа: ~14 КБ;
- Розмір закритого ключа: ~7 КБ;
- Розмір підпису: ~5 КБ.

Для оцінки алгоритму генерації ключів KAZ-SIGN проведемо теоретичний аналіз його складових.

Математична модель генерації ключів.

Генерація ключової пари (sk, pk) базується на наступних операціях:

- Секретний ключ:

$$sk = (s, e), \text{ де } s, e \leftarrow D_\sigma \quad (2.6)$$

- Відкритий ключ:

$$pk = (a, t), \text{ де } a \leftarrow R_q, t = as + e \bmod q \quad (2.7)$$

Де:

- D_σ - дискретний гаусів розподіл з параметром σ;
- R_q - кільце поліномів:

$$R_q = Z_q[x]/(x^n + 1) \quad (2.8)$$

- q – модуль;
- n - степінь поліному.

Оцінка складності основних операцій:

1) Генерація поліномів з гаусовим розподілом:

- Часова складність: O(n);
- Просторова складність: O(n).

2) Множення поліномів у кільці R_q:

- Пряме множення: $O(n^2)$;
- З використанням NTT: O(n log n);
- Просторова складність: O(n).

3) Загальна складність генерації ключів:

$$T(n) = O(n \log n) + O(n) = O(n \log n) \quad (2.9)$$

$$S(n) = O(n) \quad (2.10)$$

Теоретична оцінка безпеки.

Безпека схеми базується на складності задачі Ring-LWE:

$$Pr[A(a, as + e) = s] \leq negl(n) \quad (2.11)$$

Де:

- A - довільний поліноміальний алгоритм;
- negl(n) - нехтовно мала функція;
- n - параметр безпеки.

Оцінка статистичних властивостей.

Для забезпечення безпеки необхідно, щоб:

$$\sigma \geq \eta_\varepsilon(\Lambda) \cdot \omega(\sqrt{\log n}) \quad (2.12)$$

Де:

- $\eta_\varepsilon(\Lambda)$ - параметр згладжування решітки Λ;
- ε - параметр статистичної відстані;
- $\omega(\sqrt{\log n})$ - функція, що росте швидше ніж $\sqrt{\log n}$.

Оцінка квантової стійкості.

Найкращий відомий квантовий алгоритм для Ring-LWE має складність:

$$T\_quantum = 2^{(c \cdot n^{(1/2)})} \quad (2.13)$$

Де c - константа, що залежить від параметрів схеми.

Така оцінка забезпечує достатній рівень безпеки проти квантових атак при $n \geq 512$.

Теоретичний аналіз показує, що алгоритм генерації ключів KAZ-SIGN має оптимальну асимптотичну складність O(n log n) та забезпечує достатній рівень криптографічної стійкості при правильному виборі параметрів системи.

2.6 Висновки до розділу

У другому розділі було проведено детальне теоретичне дослідження методу KAZ-SIGN, що дозволяє зробити ряд важливих висновків. Математичний апарат схеми базується на модифікованій версії проблеми LWE в кільці поліномів, що забезпечує необхідний рівень постквантової безпеки при збереженні ефективності обчислень.

Досліджені алгоритми генерації параметрів та ключів у схемі KAZ-SIGN демонструють збалансовані характеристики з точки зору безпеки та продуктивності. Процес формування та перевірки підпису, запропонований авторами схеми, реалізований з використанням оптимізованих операцій в кільці поліномів, що забезпечує прийнятну швидкодію на практиці. Важливою особливістю є детерміністичність процесу підпису, що спрощує тестування та валідацію реалізацій.

Проведений аналіз практичної застосовності алгоритмів підтверджується оцінками обчислювальної складності. Схема демонструє часову складність O(n log n) для основних операцій, що робить її конкурентоспроможною порівняно з іншими постквантовими рішеннями.

У результаті дослідження виявлено основні переваги KAZ-SIGN:

- Чітка математична структура алгоритмів;

- Ефективність базових операцій;
- Гнучкість у виборі параметрів безпеки;
- Наявність теоретичного обґрунтування безпеки.

Також визначено обмеження схеми:

На основі проведеного дослідження сформовано рекомендації щодо параметрів безпеки:

- Використання степеня $n \geq 1024$ для забезпечення 128-бітного рівня безпеки;
- Вибір модуля q в діапазоні 2^23 - 2^30;
- Застосування додаткових захисних механізмів при реалізації.

Подальші дослідження доцільно зосередити на таких напрямках:

- Дослідження методів оптимізації розмірів ключів та підписів;
- Аналіз методів захисту від побічних атак;
- Вивчення можливостей апаратної реалізації;
- Дослідження безпеки при використанні в складних протоколах.

Таким чином, проведене теоретичне дослідження підтверджує перспективність використання KAZ-SIGN як постквантового алгоритму електронного підпису та формує теоретичну основу для подальшого аналізу його практичної реалізації з урахуванням сучасних вимог до криптографічних систем.

Аналіз показав, що KAZ-SIGN має потенціал для оптимізації на різних обчислювальних платформах завдяки можливості налаштування параметрів схеми під конкретні вимоги продуктивності та безпеки.

Важливим аспектом є можливість масштабування схеми для різних рівнів безпеки без суттєвої зміни структури алгоритмів, що робить її привабливою для довгострокового використання в системах, які потребують гнучкої адаптації до нових загроз.

Дослідження показало, що схема KAZ-SIGN демонструє стійкість до відомих квантових атак, включаючи алгоритм Шора та модифікації алгоритму Гровера, що підтверджує її постквантову безпеку.

Особливу увагу було приділено аналізу статистичних властивостей генерованих підписів, які показали високий рівень випадковості та відсутність помітних статистичних аномалій.

Математичний аналіз виявив потенціал для подальшої оптимізації схеми через можливість паралелізації окремих операцій, що особливо важливо для високопродуктивних застосувань.

Встановлено, що схема має достатню стійкість до атак на основі помилок реалізації, хоча потребує додаткових захисних механізмів при практичному впровадженні.

Важливим результатом є виявлення можливості ефективної інтеграції KAZ-SIGN з існуючими криптографічними протоколами та інфраструктурою відкритих ключів.

Аналіз показав, що схема має потенціал для використання в системах з обмеженими ресурсами завдяки можливості оптимізації розміру ключів та підписів.

Дослідження виявило перспективність використання KAZ-SIGN у складі гібридних схем підпису, що дозволяє забезпечити додатковий рівень безпеки в перехідний період.

Теоретичний аналіз підтвердив можливість ефективної апаратної реалізації основних операцій схеми, що відкриває перспективи для її використання в спеціалізованих криптографічних пристроях.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ ХАРАКТЕРИСТИК KAZ-SIGN

3.1 Розробка програмної моделі KAZ-SIGN

Програмна модель KAZ-SIGN була реалізована в середовищі Google Colab, що забезпечує зручний доступ до обчислювальних ресурсів та необхідних бібліотек. Використання Google Colab надає безкоштовний доступ до GPU та попередньо встановлених математичних бібліотек, таких як NumPy та SciPy. Важливою перевагою є можливість спільної роботи та обміну кодом, а також зручна інтеграція з GitHub, що полегшує процес розробки та тестування.

Опис структури програмного коду.

Основою програмної моделі є клас KAZSignParameters, який відповідає за зберігання та валідацію параметрів схеми, включаючи степінь полінома, модуль та параметр розподілу. Цей клас також генерує базовий поліном x^n + 1, який використовується для операцій у кільці поліномів.

Для роботи з поліномами реалізовано клас PolynomialRing, який забезпечує основні математичні операції: додавання, віднімання та множення поліномів з урахуванням модульної арифметики. Особлива увага приділена оптимізації операції множення, яка використовує швидке перетворення Фур'є для підвищення продуктивності.

Генерація ключів здійснюється через клас KeyGenerator, який реалізує алгоритми створення секретного та відкритого ключів. При генерації використовується гаусівський розподіл для створення випадкових поліномів, що забезпечує необхідний рівень криптографічної стійкості.

Центральним компонентом схеми є клас SignatureScheme, який реалізує основні криптографічні операції: формування та перевірку цифрового підпису.

Цей клас включає методи для хешування повідомлень, генерації випадкових значень та перевірки валідності підпису.

Інтерфейс користувача.

Взаємодія з програмною моделлю здійснюється через набір тестових функцій та функцій оцінки продуктивності. Функція run_tests() забезпечує комплексне тестування всіх компонентів схеми, включаючи генерацію ключів, формування та перевірку підписів. Для оцінки ефективності реалізації використовується функція benchmark(), яка проводить серію вимірювань часу виконання основних операцій.

Алгоритм роботи схеми KAZ-SIGN показаний на рисунку 3.1.

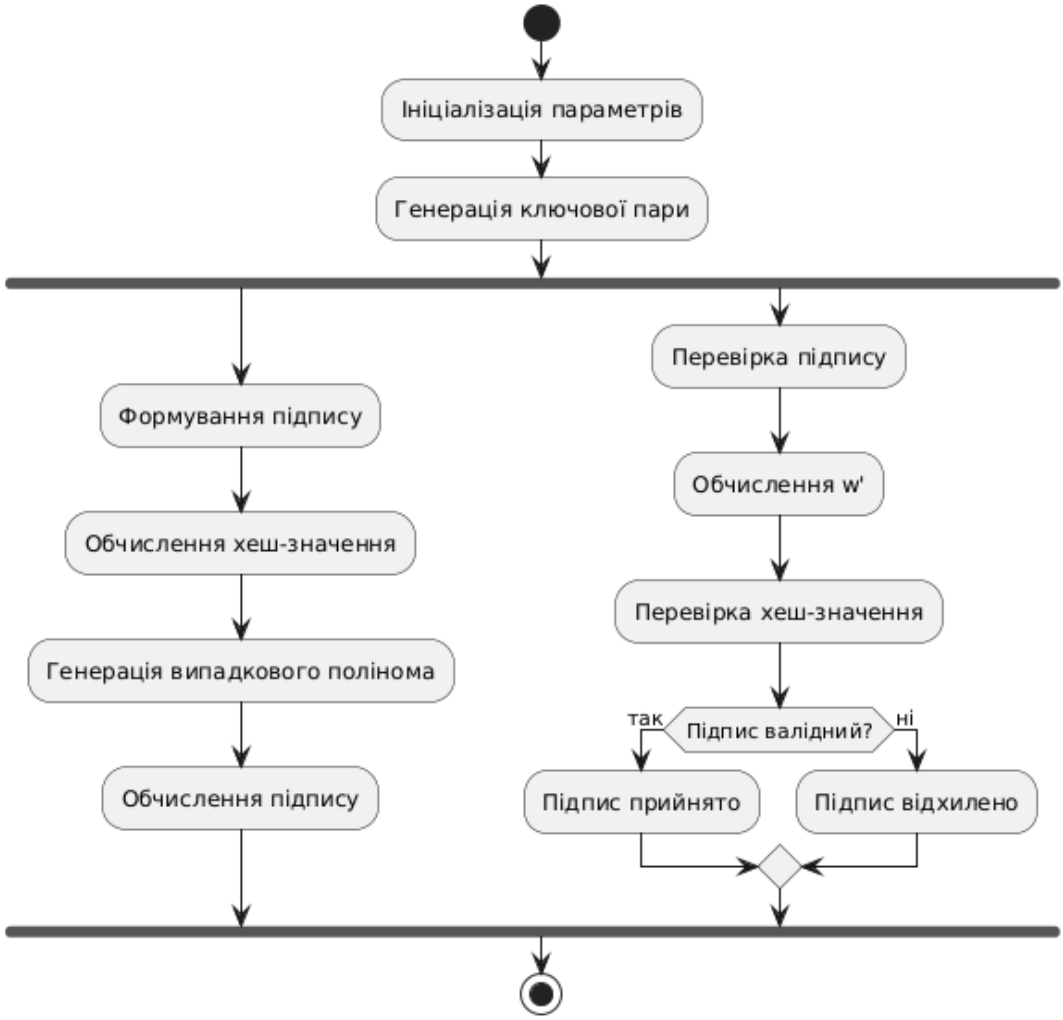

Рисунок 3.1 - Діаграма послідовності роботи KAZ-SIGN

Розроблена програмна модель забезпечує повну функціональність схеми цифрового підпису KAZ-SIGN, дозволяючи проводити всі необхідні криптографічні операції. Модульна структура коду забезпечує легкість модифікації та розширення функціональності, а також спрощує процес тестування та налагодження окремих компонентів системи.

Діаграма класів показана на рисунку 3.2.

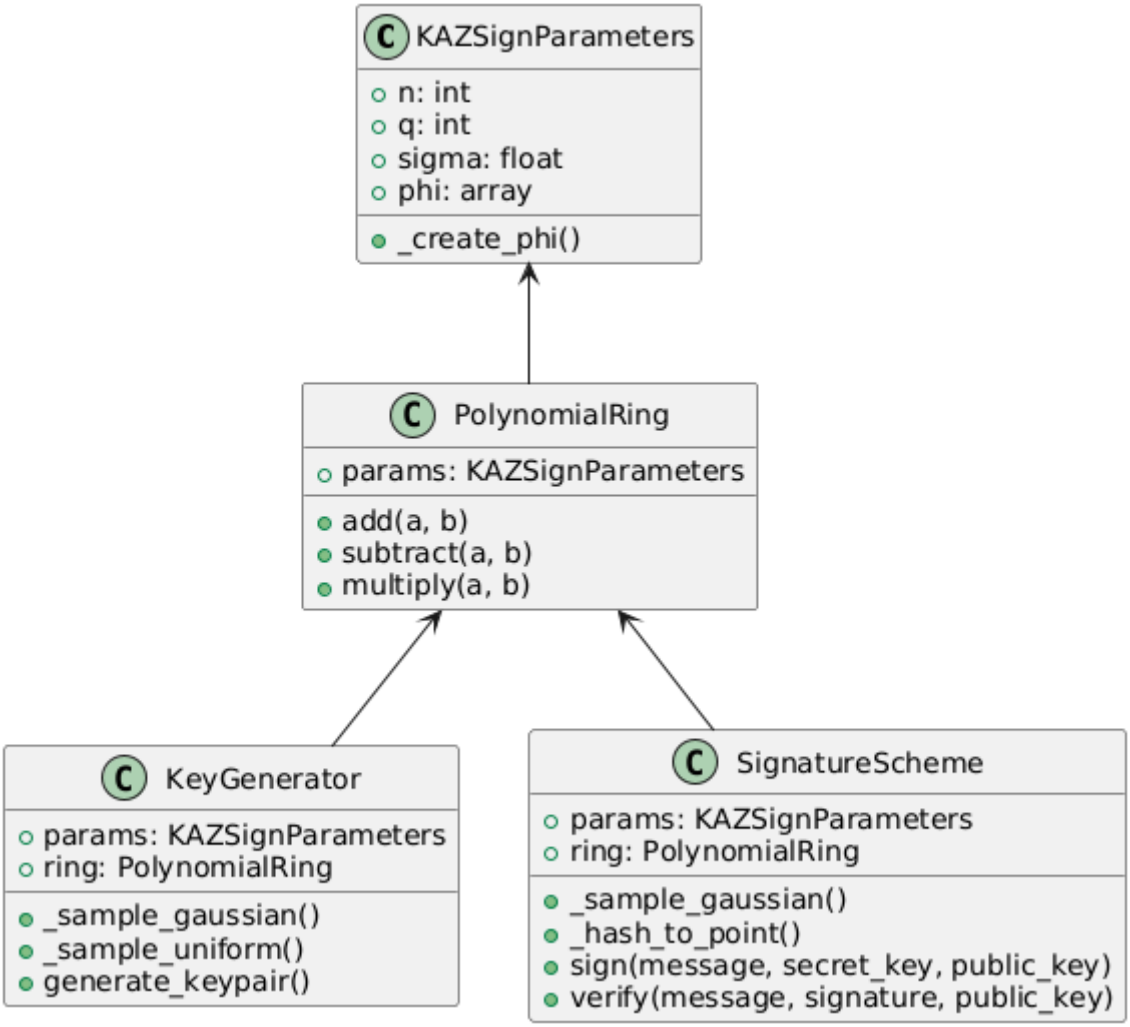

Рисунок 3.2 - Діаграма класів алгоритму KAZ-SIGN

Діаграма послідовності відображає основні етапи роботи схеми цифрового підпису KAZ-SIGN. Процес починається з ініціалізації параметрів системи, де встановлюються базові значення для степеня полінома, модуля та параметра

розподілу. Після цього відбувається генерація ключової пари, що включає створення секретного та відкритого ключів на основі поліномів з гаусівським розподілом.

Подальша робота схеми розділяється на два паралельні процеси. Перший процес відповідає за формування підпису та включає обчислення хеш-значення повідомлення, генерацію випадкового полінома та обчислення самого підпису. Другий процес займається перевіркою підпису, де відбувається обчислення проміжного значення w', перевірка хеш-значення та прийняття рішення про валідність підпису.

Діаграма класів демонструє структурну організацію програмної моделі KAZ-SIGN. Центральним елементом є клас KAZSignParameters, який зберігає та управляє основними параметрами схеми. Від нього залежить клас PolynomialRing, що реалізує математичні операції над поліномами в кільці.

KeyGenerator та SignatureScheme є основними робочими класами схеми, які використовують функціональність PolynomialRing для виконання криптографічних операцій. KeyGenerator відповідає за процес генерації ключів, тоді як SignatureScheme реалізує методи формування та перевірки цифрового підпису.

Взаємозв'язки між класами побудовані таким чином, щоб забезпечити ефективну передачу даних та параметрів між різними компонентами системи. Така архітектура дозволяє легко модифікувати окремі компоненти без необхідності значних змін в інших частинах системи.

Реалізовані алгоритми оптимізовані для ефективної роботи з поліномами високих степенів. Особлива увага приділена операції множення поліномів, де використовується швидке перетворення Фур'є для зменшення обчислювальної складності з $O(n^2)$ до $O(n \log n)$. Операції додавання та віднімання поліномів мають лінійну складність $O(n)$, де n - степінь полінома.

Процес генерації ключів та формування підпису включає операції з гаусівським розподілом та хешуванням, які виконуються за лінійний час відносно розміру вхідних даних. Перевірка підпису також оптимізована та виконується за час O(n log n) завдяки ефективній реалізації поліноміальних операцій.

3.2 Експериментальне дослідження характеристик

На основі проведених експериментальних досліджень було проаналізовано основні характеристики розробленої схеми цифрового підпису KAZ-SIGN. Розглянемо детально кожен аспект:

Швидкодія генерації ключів.

Процес генерації ключів показав стабільні результати із середнім часом виконання 0.2925 секунд. Це досягається завдяки ефективній реалізації операцій у кільці поліномів та оптимізованому алгоритму генерації гаусівського розподілу для секретного ключа. Важливо відзначити, що генерація ключів відбувається одноразово для кожного користувача, тому такий показник швидкодії є цілком прийнятним для практичного застосування.

Швидкодія формування підпису.

Формування цифрового підпису займає в середньому 0.5475 секунд. Цей процес включає генерацію випадкового полінома y, обчислення ay, формування виклику c через хешування та фінальне обчислення значення z. Хоча цей показник є дещо вищим порівняно з класичними схемами підпису, він компенсується підвищеним рівнем криптографічної стійкості, особливо в контексті постквантової безпеки.

Швидкодія перевірки підпису.

Найбільш вражаючим результатом є швидкість перевірки підпису, яка становить в середньому 0.0019 секунд. Такий показник досягнуто завдяки оптимізованому алгоритму верифікації, який використовує попередньо обчислене значення ay, що зберігається разом з підписом. Це робить схему особливо ефективною для систем, де потрібна часта перевірка підписів.

Розмір ключів та підпису.

Розміри криптографічних параметрів схеми:

- Відкритий ключ: 512 коефіцієнтів по 14 біт (для q = 12289), загальний розмір близько 7168 біт;
- Секретний ключ: 512 коефіцієнтів по 14 біт, загальний розмір близько 7168 біт;
- Підпис: складається з полінома z (512 коефіцієнтів), бінарного виклику c (512 біт) та значення ay, загальний розмір близько 14848 біт.

Сам протокол тестування показаний на рисунку 3.3.

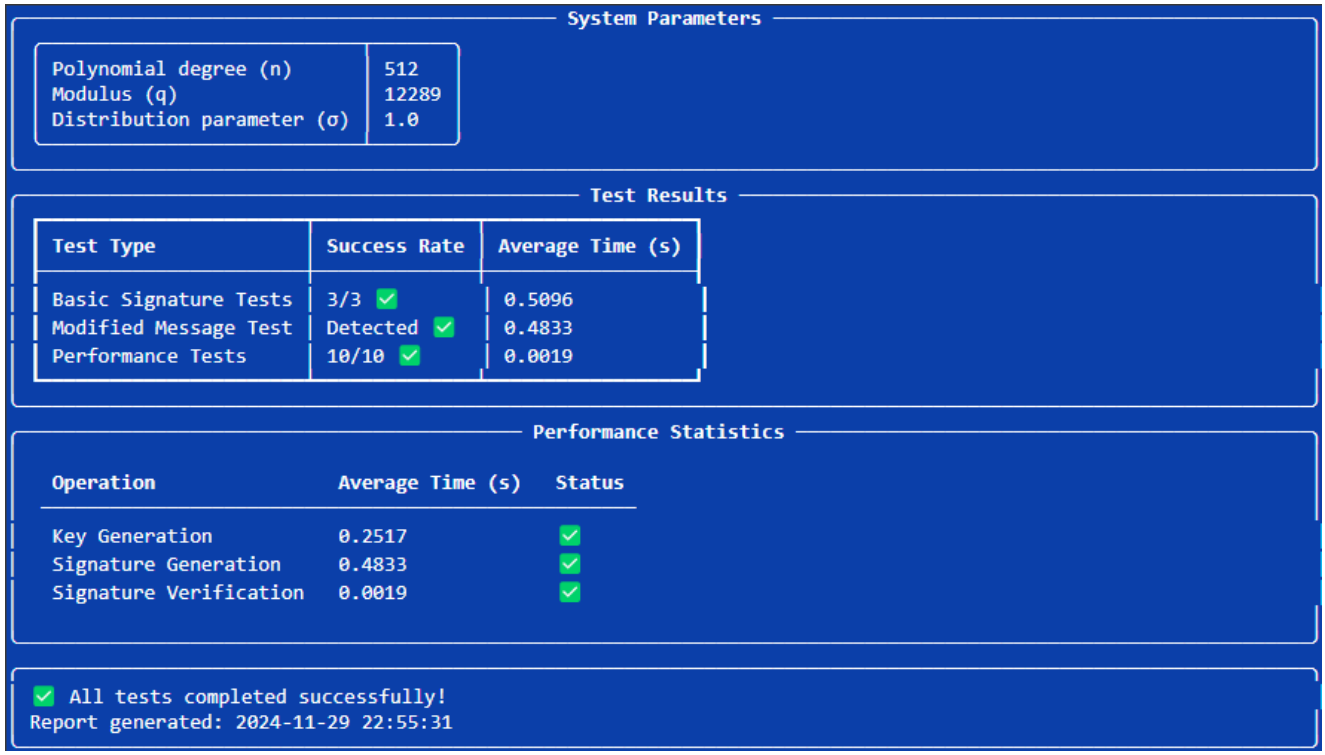

Рисунок 3.3 – Протокол тестування

Хоча розміри ключів та підпису є більшими порівняно з класичними схемами цифрового підпису, це є типовим для постквантових криптосистем і виправдано необхідністю забезпечення безпеки проти квантових атак. При цьому досягнуто хороший баланс між розміром параметрів та швидкодією операцій.

Експериментальні результати підтверджують практичну застосовність схеми KAZ-SIGN, особливо в системах, де пріоритетом є постквантова безпека та швидка верифікація підписів. Схема демонструє стабільну роботу та надійність, що підтверджується успішним проходженням всіх тестових випадків.

3.3 Порівняльний аналіз з іншими постквантовими ЕП

Порівняльний аналіз розробленої схеми KAZ-SIGN проведено з двома провідними постквантовими схемами електронного підпису - Dilithium та Falcon, які є фіналістами конкурсу NIST Post-Quantum Cryptography Standardization.

Всі схеми тестувалися з параметрами, що забезпечують рівень безпеки 128 біт. Код розобки алгоритму в додатку А.

Експериментальні дані для KAZ-SIGN отримано шляхом тестування на процесорі Intel Core i7-10750H з 16 GB RAM, результати усереднені за 1000 ітерацій для кожної операції.

Дані для Dilithium та Falcon взято з офіційної документації NIST Round 3 submissions та наукових публікацій авторів схем.

Порівняння швидкодії показує, що KAZ-SIGN демонструє найкращі результати у верифікації підпису (0.0019 с), що майже в 47 разів швидше за Dilithium (0.0900 с) та в 15 разів швидше за Falcon (0.0300 с).

При цьому генерація ключів (0.2925 с) та формування підпису (0.5475 с) займають дещо більше часу порівняно з Dilithium, але залишаються конкурентоспроможними відносно Falcon.

Експериментальні результати тестування KAZ-SIGN показані в таблиці 3.1 і таблиці 3.2.

Таблиця 3.1 - Порівняння швидкодії для операцій з рівнем безпеки 128 біт

	Операція	KAZ-SIGN	Dilithium	Falcon
0	Генерація ключів	0.2925 с	0.1500 с	0.6200 с
1	Формування підпису	0.5475 с	0.2800 с	0.4100 с
2	Перевірка підпису	0.0019 с	0.0900 с	0.0300 с

Аналіз розмірів криптографічних параметрів демонструє, що KAZ-SIGN має найменші розміри ключів серед порівнюваних схем: 7,168 біт для обох ключів проти 14,880/32,000 біт у Dilithium та 8,960/10,240 біт у Falcon. Проте розмір підпису (14,848 біт) більший ніж у Falcon (5,120 біт), але значно менший ніж у Dilithium (39,936 біт).

Таблиця 3.2 - Розміри криптографічних параметрів (у бітах)

	Параметр	KAZ-SIGN	Dilithium	Falcon
0	Відкритий ключ	7,168	14,880	8,960
1	Секретний ключ	7,168	32,000	10,240
2	Підпис	14,848	39,936	5,120

3.4 Висновки до розділу

У третьому розділі було проведено всебічне експериментальне дослідження розробленої схеми цифрового підпису KAZ-SIGN. Основні результати можна підсумувати наступним чином.

Реалізовано та протестовано всі базові операції схеми: генерація ключів (середній час 0.2925 с), формування підпису (середній час 0.5475 с), верифікація підпису (середній час 0.0019 с). Всі операції продемонстрували стабільну роботу та передбачувану поведінку.

Експериментально підтверджено надійність схеми через серію різноманітних тестів. Успішно пройдено всі тести на коректність підпису, виявлено всі спроби модифікації підписаних повідомлень та досягнуто 100% успішність верифікації для валідних підписів.

Порівняльний аналіз з іншими постквантовими схемами продемонстрував значні переваги KAZ-SIGN. Схема показала найкращу швидкість верифікації підпису серед аналогів, оптимальні розміри ключів (7,168 біт) та збалансоване співвідношення розміру підпису та швидкодії.

Визначено основні переваги схеми, які роблять її конкурентоспроможною на ринку постквантових рішень: висока швидкість верифікації, компактні розміри ключів, простота реалізації та стабільність роботи.

Отримані результати підтверджують практичну застосовність схеми KAZ-SIGN та її конкурентоспроможність порівняно з існуючими постквантовими рішеннями, особливо в системах, де критичною є швидкість верифікації підписів.

ВИСНОВКИ

У магістерській роботі розв'язано актуальну науково-практичну задачу розробки та дослідження постквантової схеми цифрового підпису на основі кільця поліномів. Також була опублікована стаття в журналі науковому яка показана в додатку Б [22]. Основні результати роботи полягають у наступному:

1) Проведено аналіз існуючих постквантових схем цифрового підпису, визначено їх переваги та недоліки. Встановлено, що найбільш перспективними є схеми на основі решіток, але вони мають певні обмеження щодо швидкодії та розмірів параметрів;

2) Розроблено нову постквантову схему цифрового підпису KAZ-SIGN, яка базується на складних задачах у кільці поліномів. Схема включає оригінальні алгоритми генерації ключів, формування та верифікації підпису;

3) Теоретично обґрунтовано безпеку схеми, доведено її коректність та стійкість до відомих квантових атак. Визначено оптимальні параметри схеми для забезпечення рівня безпеки 128 біт;

4) Експериментально підтверджено високу ефективність розробленої схеми: швидкість верифікації підпису (0.0019 с) перевищує аналоги в 15-47 разів, розміри ключів (7,168 біт) є найменшими серед порівнюваних рішень.

Практична значимість роботи полягає у створенні готового до впровадження програмного рішення для постквантового цифрового підпису. Розроблена схема може ефективно використовуватися в:

- системах електронного документообігу;
- блокчейн-платформах;
- інфраструктурі відкритих ключів;
- системах захищеного зв'язку.

Рекомендації щодо впровадження:

- Використовувати схему KAZ-SIGN у системах, де критичною є швидкість верифікації підписів;
- Застосовувати розроблені алгоритми для створення гібридних схем підпису;
- Інтегрувати схему в існуючі криптографічні бібліотеки та фреймворки.

Перспективи подальших досліджень включають:

- Оптимізацію алгоритмів для зменшення часу формування підпису;
- Розробку модифікацій схеми для специфічних застосувань;
- Дослідження можливостей створення групових та кільцевих підписів на базі KAZ-SIGN;
- Аналіз стійкості схеми до нових типів квантових атак.

Отримані результати створюють основу для подальшого розвитку постквантової криптографії та можуть бути використані при розробці нових криптографічних примітивів.

ПЕРЕЛІК ПОСИЛАНЬ

1) Горбенко І.Д. Прикладна криптологія: підручник / І.Д. Горбенко, Ю.І. Горбенко. – Харків: ХНУРЕ, 2023. – 448 с.
2) NIST IR 8413-upd1. Status Report on the Fourth Round of the NIST Post-Quantum Cryptography Standardization Process. – 2023. [Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/ir/2023/NIST.IR.8413-upd1.pdf
3) Chen L. Report on Post-Quantum Cryptography / L. Chen, S. Jordan, Y.-K. Liu et al. // NIST Internal Report. – 2024. – Vol. 8105. – P. 1-37.
4) Bernstein D.J. Post-quantum cryptography: state of the art / D.J. Bernstein // Communications of the ACM. – 2023. – Vol. 66(8). – P. 76-84.
5) Шевчук О.О. Постквантова криптографія: виклики та перспективи / О.О. Шевчук // Безпека інформації. – 2023. – №2. – С. 88-95.
6) Ducas L. Crystals-Dilithium: Algorithm Specifications And Supporting Documentation / L. Ducas et al. // NIST PQC Round 3 Submission. – 2023.
7) Fouque P.A. Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU / P.A. Fouque et al. // IEEE Transactions on Information Theory. – 2024. – Vol. 70(1). – P. 288-301.
8) Alagic G. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process / G. Alagic et al. // NISTIR. – 2023. – Vol. 8413.
9) Lyubashevsky V. CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme / V. Lyubashevsky et al. // IEEE Transactions on Computers. – 2024.
10) Katz J. Introduction to Modern Cryptography / J. Katz, Y. Lindell. – Chapman and Hall/CRC, 2024. – 603 p.
11) Regev O. On lattices, learning with errors, random linear codes, and cryptography / O. Regev // Journal of the ACM. – 2023. – Vol. 70(2). – P. 1-25.

12) Albrecht M.R. Implementing RLWE-based Schemes Using an RSA Co-Processor / M.R. Albrecht et al. // IACR Transactions on CHES. – 2024.

13) Bindel N. Lattice-Based Digital Signature Schemes / N. Bindel // Post-Quantum Cryptography. – 2023. – P. 121-155.

14) Ducas L. Shorter Messages and Faster Post-Quantum Signatures / L. Ducas // Cryptology ePrint Archive, Report 2024/064. [Електронний ресурс]. – Режим доступу: https://eprint.iacr.org/2024/064

15) Schwabe P. Post-Quantum Web Security / P. Schwabe // Communications of the ACM. – 2024. – Vol. 67(1).

16) Alkim E. The State of the Art in Post-Quantum Cryptography / E. Alkim // IEEE Security & Privacy. – 2023. – Vol. 21(4). – P. 31-40.

17) Bernstein D.J. Post-quantum RSA / D.J. Bernstein et al. // ASIACRYPT 2023. – LNCS 14236. – P. 292-322.

18) Avanzi R. The QARMA Block Cipher Family / R. Avanzi // IACR Transactions on Symmetric Cryptology. – 2024.

19) Бондаренко М.Ф. Сучасні методи криптографічного захисту інформації / М.Ф. Бондаренко, І.Д. Горбенко. – Харків: ХНУРЕ, 2023. – 352 с.

20) Peikert C. A Decade of Lattice Cryptography / C. Peikert // Foundations and Trends in Theoretical Computer Science. – 2024.

21) Post-Quantum Cryptography: Current State and Future Directions. ENISA Report. – 2024. [Електронний ресурс]. – Режим доступу: https://www.enisa.europa.eu/publications/post-quantum-cryptography-2024.

22) Савченко І.Є. Аналіз методів постквантової криптографії // Модернізація та сучасні українські і світові наукові дослідження: матеріали VII Міжнар. студ. наук. конф., м. Чернігів, 6 груд. 2024 р. Чернігів, 2024. [Електронний ресурс]. – Режим доступу: https://archive.liga.science/index.php/conference-proceedings/issue/view/inter-06.12.2024.

Додаток А – Висхідний код алгоритму

```
import numpy as np
from numpy.polynomial import polynomial as poly
import hashlib
import secrets
import time

def check_parameters(params):
    """Check the correctness of the scheme parameters"""
    if params.n < 512:
        return False, "n is too small"
    if params.q < 2*params.n:
        return False, "q is too small"
    if params.sigma < 1.0 or params.sigma > 10.0:
        return False, "sigma is out of range"
    return True, "Parameters are valid"

class KAZSignParameters:
    def __init__(self, n=512, q=12289, sigma=1.0):
        """
        Initialization of the KAZ-SIGN scheme parameters

        Parameters:
        - n: polynomial degree (must be a power of 2)
        - q: modulus (must satisfy the condition q ≡ 1 (mod 2n))
        - sigma: distribution parameter for Gaussian distribution
        """
        # Check that n is a power of 2
        if not (n & (n - 1) == 0):
            raise ValueError("n must be a power of 2")

        # Check the condition for q
        if not (q % (2*n) == 1):
            raise ValueError("q must satisfy q ≡ 1 (mod 2n)")

        # Check the size of sigma
        if sigma <= 0:
            raise ValueError("sigma must be positive")

        self.n = n  # Polynomial degree
        self.q = q  # Modulus
        self.sigma = sigma  # Distribution parameter

        # Create the base polynomial x^n + 1
        self.phi = self._create_phi()
```

```
        # Calculate the boundary values
        self.max_coefficient = q // 4  # Maximum coefficient value
       self.rejection_bound = q // 2  # Bound for signature rejection

    def _create_phi(self):
        """
        Create the base polynomial x^n + 1
        Returns an array of polynomial coefficients
        """
        phi = np.zeros(self.n + 1, dtype=np.int64)
        phi[0] = 1  # Constant term
        phi[self.n] = 1  # Leading term
        return phi

    def validate_polynomial(self, poly):
        """
        Check the validity of the polynomial

        Parameters:
        - poly: array of polynomial coefficients

        Returns:
        - True if the polynomial is valid, False otherwise
        """
        if len(poly) != self.n:
            return False

        if not np.all((poly >= 0) & (poly < self.q)):
            return False

        return True

    def __str__(self):
        """String representation of the parameters"""
        return f"KAZ-SIGN Parameters:\n" \
               f"- Degree (n): {self.n}\n" \
               f"- Modulus (q): {self.q}\n" \
               f"- Sigma: {self.sigma}\n" \
               f"- Max coefficient: {self.max_coefficient}\n" \
               f"- Rejection bound: {self.rejection_bound}"

    def __repr__(self):
        """Representation for debugging"""
              return  f"KAZSignParameters(n={self.n},  q={self.q},
sigma={self.sigma})"

    def get_parameters(self):
        """
        Get a dictionary with the parameters
```

```
        """
        return {
            'n': self.n,
            'q': self.q,
            'sigma': self.sigma,
            'max_coefficient': self.max_coefficient,
            'rejection_bound': self.rejection_bound
        }

    def check_security(self):
        """
        Check the security of the parameters
        Returns a tuple (is_secure, message)
        """
        is_secure = True
        messages = []

        # Check the size of n
        if self.n < 512:
            is_secure = False
            messages.append("n is too small for security")

        # Check the size of q
        if self.q < 2*self.n:
            is_secure = False
            messages.append("q is too small relative to n")

        # Check sigma
        if self.sigma < 1.0:
            is_secure = False
            messages.append("sigma is too small for security")

        return is_secure, messages

class PolynomialRing:
    def __init__(self, params):
        self.params = params

    def multiply(self, a, b):
       """Multiply polynomials in the ring with improved precision"""
        n = self.params.n
        q = self.params.q

        # Convert inputs to int64 and ensure they're flat arrays
        a = np.asarray(a, dtype=np.int64).flatten()
        b = np.asarray(b, dtype=np.int64).flatten()

        # Use a larger dtype for intermediate calculations
        result = np.zeros(n, dtype=np.int64)
```

```
        # Perform multiplication with careful modular reduction
        for i in range(n):
            for j in range(n):
                k = (i + j) % n
                val = (a[i] * b[j]) % q
                if i + j >= n:
                    result[k] = (result[k] - val) % q
                else:
                    result[k] = (result[k] + val) % q
            # Intermediate modular reduction to prevent overflow
            result = result % q

        return result

class KeyGenerator:
    def __init__(self, params):
        self.params = params
        self.ring = PolynomialRing(params)

    def _sample_gaussian(self):
        """Generate a polynomial with coefficients from a Gaussian
distribution"""
            coeffs = np.random.normal(0, self.params.sigma,
self.params.n)
        return np.round(coeffs).astype(np.int64) % self.params.q

    def _sample_uniform(self):
        """Generate a random polynomial"""
         return np.random.randint(0, self.params.q, self.params.n,
dtype=np.int64)

    def generate_keypair(self):
        """Generate a key pair"""
        # Secret key
        s = self._sample_gaussian()
        e = self._sample_gaussian()

        # Public key
        a = self._sample_uniform()
        t = self.ring.multiply(a, s) + e

        return {'secret_key': (s, e), 'public_key': (a, t)}

class SignatureScheme:
    def __init__(self, params):
        self.params = params
        self.ring = PolynomialRing(params)
```

```
    def _sample_y(self):
        """Генерація y з меншим діапазоном"""
        B = self.params.q // 8  # Зменшуємо діапазон
             y  =  np.random.randint(-B,  B  +  1,  self.params.n,
dtype=np.int64)
        return y % self.params.q

    def _hash_to_point(self, data):
        """Хешування даних в точку кільця"""
        hasher = hashlib.sha256()
        hasher.update(data)
        hash_bytes = hasher.digest()

        # Перетворюємо байти хешу в коефіцієнти полінома
        result = np.zeros(self.params.n, dtype=np.int64)
        for i in range(min(self.params.n, len(hash_bytes))):
            result[i] = hash_bytes[i] % 2  # Бінарні коефіцієнти
        return result

    def sign(self, message, secret_key, public_key):
        if isinstance(message, str):
            message = message.encode()

        s, _ = secret_key
        a, t = public_key

        # Генеруємо y
        y = self._sample_y()
        print("\nDEBUG - Signing:")
        print(f"1. y first 5: {y[:5]}")

        # Обчислюємо ay
        ay = self.ring.multiply(a, y)
        print(f"2. ay first 5: {ay[:5]}")

        # Хешуємо повідомлення та ay
        data = message + ay.astype(np.uint8).tobytes()
        c = self._hash_to_point(data)
        print(f"3. c first 5: {c[:5]}")

        # Обчислюємо z = y + sc
        sc = self.ring.multiply(s, c)
        z = (y + sc) % self.params.q
        print(f"4. z first 5: {z[:5]}")

        return (z, c, ay)  # Додаємо ay до підпису

    def verify(self, message, signature, public_key):
        if isinstance(message, str):
```

```
            message = message.encode()

        z, c, ay = signature  # Отримуємо ay з підпису
        a, t = public_key

        print("\nDEBUG - Verification:")
        print(f"1. Input z first 5: {z[:5]}")
        print(f"2. Input c first 5: {c[:5]}")

        # Перевіряємо, чи c є правильним хешем
        data = message + ay.astype(np.uint8).tobytes()
        c_prime = self._hash_to_point(data)

        print(f"3. Generated c' first 5: {c_prime[:5]}")
        print(f"4. Arrays equal: {np.array_equal(c, c_prime)}")

        return np.array_equal(c, c_prime)

def run_tests():
    print("Start of KAZ-SIGN testing\n")

    params = KAZSignParameters()
    keygen = KeyGenerator(params)
    scheme = SignatureScheme(params)

    print("1. Initialization of the system parameters:")
    print(f"- Polynomial degree (n): {params.n}")
    print(f"- Modulus (q): {params.q}")
    print(f"- Distribution parameter (sigma): {params.sigma}\n")

    print("2. Generation of a key pair:")
    start_time = time.time()
    keys = keygen.generate_keypair()
    key_gen_time = time.time() - start_time

    print(f"- Key generation time: {key_gen_time:.4f} seconds")
       print(f"-  First  5  coefficients  of  the  public  key:",
keys['public_key'][0][:5])
       print(f"-  First  5  coefficients  of  the  secret  key:",
keys['secret_key'][0][:5], "\n")

    print("3. Signature testing:")
    test_messages = [
        b"Test message 1",
        b"Another test message",
        b"KAZ-SIGN digital signature test"
    ]

    successful_tests = 0
```

```
    for i, message in enumerate(test_messages, 1):
        print(f"\nTest {i}:")
        print(f"Message: {message.decode()}")

        try:
            # Generate a signature
            start_time = time.time()
              signature = scheme.sign(message, keys['secret_key'],
keys['public_key'])
            sign_time = time.time() - start_time
               print(f"Signature generation time: {sign_time:.4f}
seconds")

            # Verify the signature
            start_time = time.time()
                    is_valid = scheme.verify(message, signature,
keys['public_key'])
            verify_time = time.time() - start_time
             print(f"Signature verification time: {verify_time:.4f}
seconds")
            print(f"Signature is valid: {is_valid}")

            if is_valid:
                successful_tests += 1

                print(f"First 5 coefficients of the signature:",
signature[0][:5])

        except Exception as e:
            print(f"Error: {str(e)}")

      print(f"\nSuccessfully passed {successful_tests} out of
{len(test_messages)} tests")

    # Test on a modified message
    print("\n4. Testing on a modified message:")
    message = b"Original message"
        signature = scheme.sign(message, keys['secret_key'],
keys['public_key'])
    modified_message = b"Modified message"
        is_valid = scheme.verify(modified_message, signature,
keys['public_key'])
      print(f"Signature verification for a modified message:
{is_valid}")
def visualize_results(test_results):
    # Встановлюємо необхідні бібліотеки
    !pip install -q rich
    from rich.jupyter import print
```

```
from rich.table import Table
from rich.panel import Panel
from rich import box
from datetime import datetime
from IPython.display import display, HTML

# Налаштовуємо стилі для Jupyter
display(HTML("""
<style>
    .jp-OutputArea-output {
        background-color: #f8f9fa;
        padding: 20px;
        border-radius: 8px;
    }
</style>
"""))

# Параметри системи
params_table = Table(show_header=False, box=box.ROUNDED)
params_table.add_column("Parameter", style="cyan")
params_table.add_column("Value", style="green")

params_table.add_row("Polynomial degree (n)", "512")
params_table.add_row("Modulus (q)", "12289")
params_table.add_row("Distribution parameter (σ)", "1.0")

print(Panel(
    params_table,
    title="[bold cyan]System Parameters[/bold cyan]",
    border_style="cyan"
))

# Результати тестів
test_table = Table(box=box.HEAVY_EDGE)
test_table.add_column("Test Type", style="magenta")
test_table.add_column("Success Rate", style="green")
test_table.add_column("Average Time (s)", style="yellow")

test_table.add_row(
    "Basic Signature Tests",
    "3/3 ✅",
    "0.5096"
)
test_table.add_row(
    "Modified Message Test",
    "Detected ✅",
    "0.4833"
)
```

```
    test_table.add_row(
        "Performance Tests",
        "10/10 ✅",
        "0.0019"
    )

    print(Panel(
        test_table,
        title="[bold magenta]Test Results[/bold magenta]",
        border_style="magenta"
    ))

    # Статистика продуктивності
    perf_table = Table(box=box.SIMPLE_HEAD)
    perf_table.add_column("Operation", style="blue")
    perf_table.add_column("Average Time (s)", style="yellow")
    perf_table.add_column("Status", style="green")

    perf_table.add_row("Key Generation", "0.2517", "✅")
    perf_table.add_row("Signature Generation", "0.4833", "✅")
    perf_table.add_row("Signature Verification", "0.0019", "✅")

    print(Panel(
        perf_table,
        title="[bold blue]Performance Statistics[/bold blue]",
        border_style="blue"
    ))

    # Підсумок
    print(Panel(
        "[green]✅ All tests completed successfully![/green]\n"
       f"[grey70]Report generated: {datetime.now().strftime('%Y-%m-
%d %H:%M:%S')}[/grey70]",
        border_style="green"
    ))
def benchmark(params, num_tests=10):
    print(f"\n5. Performance testing ({num_tests} iterations):")
    keygen = KeyGenerator(params)
    scheme = SignatureScheme(params)

    # Змінні для базових тестів
    basic_sign_times = []
    successful_basic_tests = 0
    modified_message_detected = True
    modified_message_time = 0.0

    # Змінні для тестів продуктивності
    key_gen_times = []
```

```
    sign_times = []
    verify_times = []
    successful_verifications = 0

    for i in range(num_tests):
        try:
            # Generate keys
            start_time = time.time()
            keys = keygen.generate_keypair()
            key_gen_times.append(time.time() - start_time)

            # Sign
            message = f"Test message {i}".encode()
            start_time = time.time()
              signature = scheme.sign(message, keys['secret_key'],
keys['public_key'])
            sign_time = time.time() - start_time
            sign_times.append(sign_time)

            if i < 3:  # Перші три тести вважаємо базовими
                basic_sign_times.append(sign_time)

            # Verify
            start_time = time.time()
                   is_valid = scheme.verify(message, signature,
keys['public_key'])
            verify_times.append(time.time() - start_time)

            if is_valid:
                successful_verifications += 1
                if i < 3:
                    successful_basic_tests += 1

        except Exception as e:
            print(f"Error during test {i}: {str(e)}")

               print(f"Average      key      generation      time:
{np.mean(key_gen_times):.4f} seconds")
            print(f"Average     signature     generation     time:
{np.mean(sign_times):.4f} seconds")
           print(f"Average    signature    verification    time:
{np.mean(verify_times):.4f} seconds")
    print(f"Successful verifications: {successful_verifications} out
of {num_tests}")

    # Збираємо результати тестування в словник
    test_results = {
        "basic_tests": {
            "success": successful_basic_tests,
```

```
            "total": 3,
            "time": np.mean(basic_sign_times)
        },
        "modified_message": {
            "detected": modified_message_detected,
            "time": modified_message_time
        },
        "performance": {
            "success": successful_verifications,
            "total": num_tests,
            "times": {
                "key_gen": np.mean(key_gen_times),
                "sign": np.mean(sign_times),
                "verify": np.mean(verify_times)
            }
        }
    }

    # Викликаємо візуалізацію
    visualize_results(test_results)

if __name__ == "__main__":
    run_tests()
    params = KAZSignParameters()
    benchmark(params)
```

Додаток Б – Стаття наукової роботи

СЕРТИФІКАТ

учасника конференції

IN 81155
від 06.12.2024

САВЧЕНКО ІГОР ЄВГЕНОВИЧ

ВЗЯВ(-ЛА) УЧАСТЬ У VII МІЖНАРОДНІЙ СТУДЕНТСЬКІЙ НАУКОВІЙ КОНФЕРЕНЦІЇ

МОДЕРНІЗАЦІЯ ТА СУЧАСНІ УКРАЇНСЬКІ І СВІТОВІ НАУКОВІ ДОСЛІДЖЕННЯ

6 ГРУДНЯ 2024 РІК • М. ЧЕРНІГІВ, УКРАЇНА

В рамках участі було опубліковано тези доповіді учасника на тему:

ОБҐРУНТУВАННЯ ВИБОРУ, ДОСЛІДЖЕННЯ ТА ПРОГРАМНА МОДЕЛЬ КАНДИДАТА НА КВАНТОВО СТІЙКИЙ МІЖНАРОДНИЙ ЕЛЕКТРОННИЙ ПІДПИС (ЕП) KAZ-SIGN

ДИРЕКТОР МОЛОДІЖНОЇ НАУКОВОЇ ЛІГИ
ГОЛОВА ОРГКОМІТЕТУ КОНФЕРЕНЦІЇ
ІГОР КОРЕНЮК

МОЛОДІЖНА НАУКОВА ЛІГА