Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Факультет комп'ютерних наук
Спеціальність 125 «Кібербезпека»
Освітня програма «Безпека інформаційних та комунікаційних систем»

«Допущено до захисту»
Зав. кафедрою БІСТ
Сергій РАССОМАХІН
_____
« » 2022 р.

Пояснювальна записка
до кваліфікаційної роботи магістра
на тему: « Науково-практичні основи оцінки та порівняння постквантових стандартів електронних підписів Dilithium та Вершина для Блокчейн »

оцінка « »
Голова ЕК
Доценко С.І. _____

Керівник проф. Горбенко І.Д.
Рецензент к.т.н. Бобух В.А.
Виконавець : студент групи КБ-61
Козюберда Д.О. _____

Харків – 2022

РЕФЕРАТ

Пояснювальна записка до проекту магістра містить 80 сторінок, 7 рисунків, 8 таблиць, 40 посилань на джерела та 3 додатки.

Мета роботи полягає в досліджені, оцінці та порівнянні постквантових стандартів електронних цифрових підписів CRYSTALS-Dilithium та Вершина зокрема для технології Блокчейн.

Об’єкт дослідження – процеси електронного цифрового підпису на основі алгебраїчних решіток та їх використання у технології Блокчейн.

Предмет дослідження – порівняння та аналіз результатів методів електронного цифрового підпису, що засновані на математиці алгебраїчних решіток, зокрема для оцінки ефективності використання в контексті Блокчейн.

Основними методами досліджень є аналіз обраних технологій електронного цифрового підпису та їх властивостей.

Завдання, яке вирішується в дипломній роботі полягає в тому, щоб розглянути та проаналізувати результати тестування електронних цифрових підписів, що засновані на алгебраїчних решітках та можуть бути використані у якості стійких постквантових алгоритмів у постквантовому світі, а зокрема в контексті технології Блокчейн.

Результати роботи можуть бути використані в подальшому дослідженні, імплементації чи покращенні продуктивності використання постквантових стандартів електронних цифрових підписів на основі алгебраїчних решіток у технології Блокчейн, окремо від цієї технології чи в інших наукових роботах. Результати роботи також сприяють дискусії про постквантові криптографічні стандарти, їх розробку, вдосконалення, імплементацію, тощо.

Ключові слова: ІНФОРМАЦІЙНА БЕЗПЕКА, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, КРИПТОГРАФІЧНІ АЛГОРИТМИ, ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС, БЛОКЧЕЙН, КРИПТОГРАФІЯ АЛГЕБРАЇЧНИХ

РЕШІТОК, МАТЕМАТИЧНА МОДЕЛЬ, ПОСТКВАНТОВА КРИПТОГРАФІЯ.

ABSTRACT

The explanatory note to the master's project contains 80 pages, 7 figures, 8 tables, 40 references to sources and 3 appendices.

The purpose of the work is to research, evaluate and compare the post-quantum standards of electronic digital signatures CRYSTALS-Dilithium and Vershyna, in particular for Blockchain technology.

The object of research is electronic digital signature processes based on algebraic lattices and their use in Blockchain technology.

The subject of the study is the comparison and analysis of the results of electronic digital signature methods based on the mathematics of algebraic lattices, in particular for evaluating the effectiveness of use in the context of Blockchain.

The main research methods are the analysis of selected electronic digital signature technologies and their properties.

The task that is solved in the thesis is to consider and analyze the results of testing electronic digital signatures, which are based on algebraic lattices and can be used as stable post-quantum algorithms in the post-quantum world, and in particular in the context of Blockchain technology.

The results of the work can be used in further research, implementation or performance improvement of the use of post-quantum standards of electronic digital signatures based on algebraic lattices in Blockchain technology, separately from this technology or in other scientific works. The results of the work also contribute to the discussion about post-quantum cryptographic standards, their development, improvement, implementation, etc.

Key words: INFORMATION SECURITY, INFORMATION TECHNOLOGY, CRYPTOGRAPHIC ALGORITHMS, ELECTRONIC DIGITAL SIGNATURE, BLOCKCHAIN, ALGEBRAIC LATTICE CRYPTOGRAPHY, MATHEMATICAL MODEL, POST-QUANTUM CRYPTOGRAPHY.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ СКОРОЧЕНЬ І ТЕРМІНІВ

ЕЦП	—	Електронний цифровий підпис
ЕП	—	Електронний підпис
ЦП	—	Центральний процесор
LWE	—	Learning with errors
MLWE	—	Module learning with errors
NTT	—	Number Theoretic Transform
ROM	—	Random Oracle model
QROM	—	Quantum Random Oracle model
NIST	—	National Institute of Standards and Technology
PK	—	Public key
SK	—	Secret key
BKZ	—	Block–Korkine–Zolotarev algorithm
OQS	—	Open Quantum Safe
DLT	—	Distributed Ledger Technology
ECDSA	—	Elliptic Curve Digital Signature Algorithm
CA	—	Certificate authority
SHA	—	Secure Hash Algorithm
MSP	—	Membership Service Provider
BCCSP	—	Blockchain Cryptographic Service Provider interface

ВСТУП

Академічні дослідження потенційного впливу квантових обчислень датуються щонайменше 2001 роком. У звіті NIST, опублікованому в квітні 2016 року, цитуються експерти, які визнають можливість використання квантової технології, що зробить вразливим алгоритм RSA, який зазвичай використовується, а статися це може до 2030 року [1]. У результаті такого роду досліджень виникла потреба стандартизувати квантово-захищені криптографічні примітиви. Оскільки більшість симетричних примітивів відносно легко модифікувати таким чином, щоб зробити їх квантово стійкими, зусилля були зосереджені на криптографії з відкритим ключем, а саме на цифрових підписах і механізмах інкапсуляції ключів. У грудні 2016 року NIST розпочав процес стандартизації, оголосивши конкурс.

Змагання зараз проходить третій раунд із очікуваних чотирьох, у кожному раунді деякі алгоритми відкидаються, а інші вивчаються більш ретельно. NIST сподівається опублікувати стандартизаційні документи до 2024 року, але може пришвидшити процес, якщо будуть зроблені серйозні прориви в квантових обчисленнях.

Наразі не вирішено, чи будуть майбутні стандарти опубліковані як FIPS чи як NIST Special Publication (SP), але вони безперечно будуть відігравати важливу роль у постквантовій криптографії.

23 схеми підпису та 59 схем шифрування/KEM були подані до кінцевого терміну подання наприкінці 2017 року [2], з яких 69 було визнано завершеними та належними та брали участь у першому раунді. Сім із них, з яких три є схемами підписів, пройшли до третього раунду, про який було оголошено 22 липня 2020 року.

Через те, що вже існуючі квантові комп’ютери постійно вдосконалюються, а процес винайдення нових також не стоїть на місці, існує загальна потреба в стандартизації постквантових електронних підписів. В

межах конкурсу NIST PQC до третього раунду пройшли 3 електронних підписи, що є найбільш перспективними та претендують на стійкість до класичного та квантового криптоаналізу та можливість їх застосування в постквантовий період. Зокрема до третього раунду увійшли такі електронні підписи як CRYSTALS-DILITHIUM, FALCON та Rainbow, що вважаються основними. Також окрім основних кандидатів було відібрано ще три альтернативних: Picnic, SPHINCS+, GeMSS.

29 вересня 2022 року було опубліковано фінальну версію звіту про стан третього раунду процесу стандартизації постквантової криптографії NIST [3].

У цьому звіті описується процес оцінки та відбору кандидатів третього раунду процесу постквантової стандартизації криптографії NIST на основі відгуків громадськості та внутрішньої перевірки. У звіті підсумовано кожен із 15 алгоритмів-кандидатів третього раунду та визначено ті, які були обрані для стандартизації, а також ті, які продовжуватимуть оцінюватися в четвертому раунді аналізу. Згідно з ци будуть стандартизовані цифрові підписи CRYSTALS–Dilithium, FALCON і SPHINCS+. Хоча вибрано кілька алгоритмів підпису, NIST рекомендує CRYSTALS–Dilithium як основний алгоритм для впровадження. Крім того, чотири з альтернативних алгоритмів-кандидатів на встановлення ключа пройдуть до четвертого раунду оцінювання: BIKE, Classic McEliece, HQC і SIKE. Ці кандидати все ще розглядаються для майбутньої стандартизації. NIST також опублікує новий конкурс пропозицій щодо алгоритмів цифрового підпису з відкритим ключем, щоб розширити та диверсифікувати свій портфель підписів.

Тож далі у даній роботі я детальніше розгляну та порівняю деякі схеми електронного підпису, такі як основну CRYSTALS–Dilithium, а проєкти національних стандартів «Вершина 1» та «Вершина 2». Зокрема дане порівняння буде проведене з огляду на перспективність використання у Блокчейн.

1 ЗАГАЛЬНІ ВІДОМОСТІ ТА АНАЛІЗ ТЕХНОЛОГІЇ ЕП CRYSTALS-DILITHIUM

Dilithium – схема цифрового підпису, безпека якої базується на складності пошуку коротких векторів у решітках. Дана схема розроблена з урахуванням таких критеріїв:

- Легко реалізувати безпечно. Найкомпактніші схеми підпису на основі решітки [4, 5] надзвичайно сильно вимагають генерації секретної випадковості з дискретного розподілу Гауса. Створення таких зразків захищеним від атак побічних каналів є дуже нетривіальним і може легко призвести до незахищених реалізацій, як показано в [6, 7, 8]. Хоча ймовірно, що дуже ретельна реалізація може запобігти таким атакам, нерозумно припускати, що універсально розгорнута схема, яка містить багато тонкощів, завжди буде реалізована експертами. Таким чином Dilithium використовує лише рівномірну вибірку, як спочатку було запропоновано для підписів у [9, 10]. Крім того, всі інші операції (такі як множення полінома та округлення) легко реалізуються в постійному часі.
- Будьте консервативними з параметрами. Оскільки розробники схеми націлені на довгострокову безпеку, вони проаналізували можливість застосування гратчастих атак (атак з використанням решіток) з дуже сприятливої для зловмисника точки зору. Зокрема, вони розглядають (квантові) алгоритми, вимоги до простору яких такого ж порядку, як і до часу. Такі алгоритми наразі нереалістичні, і, здається, існують серйозні перешкоди для усунення вимог щодо простору, але вони допускають можливість того, що в майбутньому можуть відбутися вдосконалення.

- Мінімізуйте розмір відкритого ключа та підпису. Оскільки багато програм вимагають передачі як відкритого ключа, так і підпису (наприклад, ланцюжки сертифікатів), дана схема розроблена таким чином, щоб мінімізувати суму цих параметрів. Згідно з обмеженням, що уникають (дискретної) вибірки Гауса, наскільки відомо, Dilithium має найменшу комбінацію розмірів підпису та відкритого ключа з усіх схем на основі решітки з однаковими рівнями безпеки.
- Бути модульним – легко варіювати безпеку. Дві операції, які складають майже всю процедуру підписання та перевірки, — це розширення XOF (використовується SHAKE-128 і SHAKE-256) і множення в поліноміальному кільці $\mathbb{Z}_q[X]/(X^n + 1)$. Тому високоефективні реалізації даного алгоритму повинні оптимізувати ці операції та переконатися, що вони виконуються в постійному часі. Для всіх рівнів безпеки схема використовує те саме кільце з $q = 2^{23} - 2^{13} + 1$ і $n = 256$. Змінна безпека просто передбачає виконання більше/менше операцій над цим кільцем і більше/менше розширення XOF. Іншими словами, як тільки оптимізована реалізація отримана для певного рівня безпеки, легко отримати оптимізовану реалізацію для вищого/нижчого рівня.

1.1 Загальна специфікація алгоритму

Конструкція схеми заснована на підході «Фіат-Шамір з перервами» [9, 11] і найбільше схожа на схеми, запропоновані в [10, 12]. На рисунку 1.1 представлено спрощену (і менш ефективну) версію схеми. Ця версія є дещо модифікованою версією схеми з [12]. Нижче ми розглянемо кожен із компонентів, щоб дати читачеві уявлення про те, як працюють такі схеми.

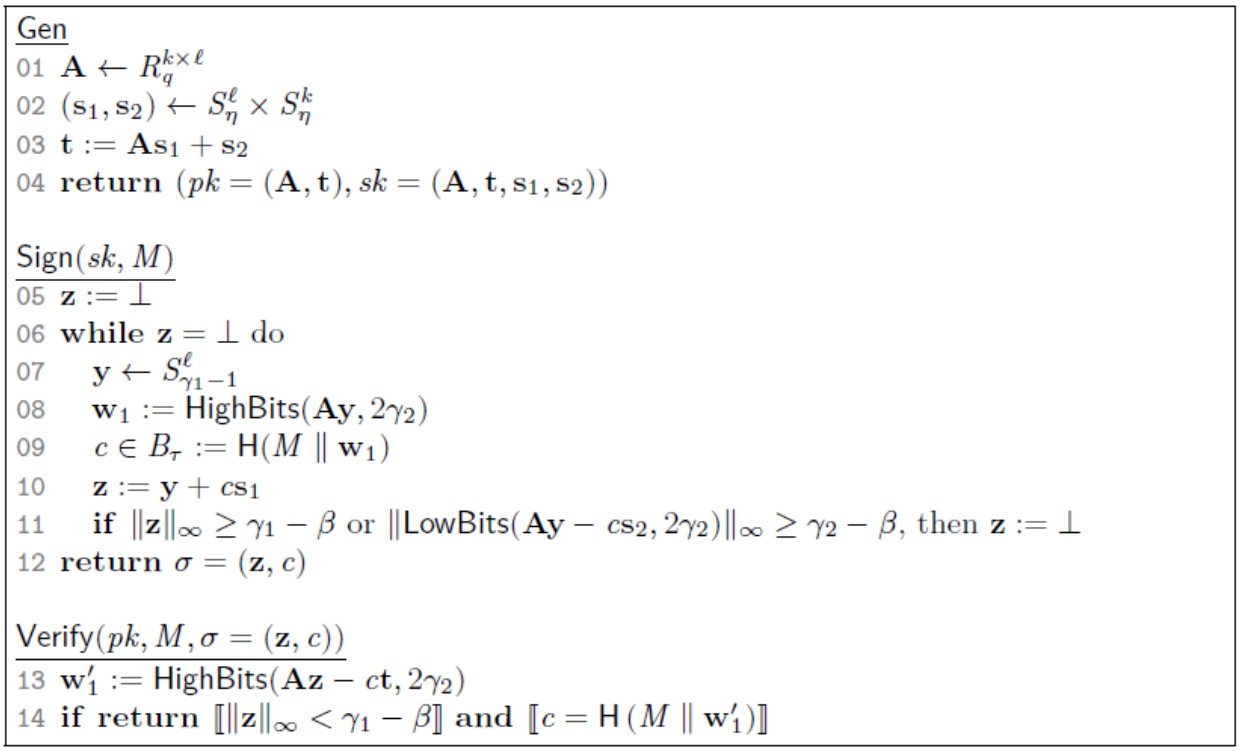

Рисунок 1.1 – Шаблон для схеми підпису, що розглядається без стиснення відкритого ключа

Генерація ключів

Алгоритм генерації ключа генерує $k \times l$ матрицю A, кожен із записів якої є поліномом у кільці $\mathrm{R_q} = \mathbb{Z}_q[X]/(X^n + 1)$. Як згадувалося раніше, ми завжди матимемо $q = 2^{23} - 2^{13} + 1$ і $n = 256$. Після цього алгоритм відбирає вектори випадкових секретних ключів s_1 і s_2. Кожен коефіцієнт цих векторів є елементом $\mathrm{R_q}$ з малими коефіцієнтами розміру не більше η. Нарешті, друга частина відкритого ключа обчислюється як $t = As_1 + s_2$. Передбачається, що всі алгебраїчні операції в цій схемі виконуються над кільцем поліномів $\mathrm{R_q}$.

Порядок підписання

Алгоритм підпису генерує маскувальний вектор поліномів y з коефіцієнтами, меншими за y_1. Параметр y_1 встановлюється стратегічно – він достатньо великий, щоб остаточний підпис не розкривав секретний ключ (тобто алгоритм підпису є нульовим знанням), але малий достатньо, щоб підпис було важко підробити. Потім підписувач обчислює Ay і встановлює w_1

як «старші» біти коефіцієнтів у цьому векторі. Зокрема, кожен коефіцієнт w в Ay можна записати канонічним способом у вигляді $w = w_1 \cdot 2\gamma_2 + w_0$, де $|w_0| \leq \gamma_2$; тоді w_1 є вектором, що містить усі w_1. Потім створюється виклик c як хеш повідомлення та w_1. Вихід с є поліномом від $\mathrm{R_q}$ із рівними $\tau \pm 1$ та рештою 0. Причина такого розподілу полягає в тому, що c має малу норму та походить із домену розміром $log_2\binom{256}{\tau} + \tau$, який хотілося б мати між 128 і 256. Тоді потенційний підпис обчислюється як $z = y + cs_1$.

Якщо z було б безпосередньо виведено в цей момент, то схема підпису була б незахищеною через те, що секретний ключ буде втрачено. Щоб уникнути залежності z від секретного ключа, використовується вибірка відхилення. Параметр β встановлюється як максимально можливий коефіцієнт cs_i. Оскільки c має $\tau \pm 1$, а максимальний коефіцієнт у s_i дорівнює η, легко побачити, що $\beta \leq \tau \cdot \eta$. Якщо будь-який коефіцієнт при z більшій за $y_1 - \beta$, тоді процедура підписання відхиляється та перезапускається. Крім того, якщо будь-який коефіцієнт молодших бітів $Az - ct$ більший за $y_2 - \beta$, також перезапускається. Перша перевірка необхідна для безпеки, тоді як друга необхідна як для безпеки, так і для правильності. Цикл while у процедурі підписання продовжує повторюватися, доки не будуть виконані дві попередні умови. Параметри встановлюються так, щоб очікувана кількість повторень не була надто високою (наприклад, близько 4).

Перевірка

Верифікатор спочатку обчислює w_1' як старші біти $Az - ct$, а потім приймає, якщо всі коефіцієнти z менші за $y_1 - \beta$ та якщо c є гешем повідомлення та w_1'. Давайте розглянемо, чому перевірка працює, зокрема, чому $HighBits(Az - ct, 2y_2) = HighBits(Ay, 2y_2)$. Перше, на що слід звернути увагу, це те, що $Az - ct = Ay - cs_2$. Отже, все, що дійсно потрібно показати це (1.1):

$$HighBits(Ay, 2y_2) = HighBits(Ay - cs_2, 2y_2) \qquad (1.1)$$

Причиною цього є те, що дійсний підпис матиме $\|LowBits(Ay - cs_2, 2y_2)\|_\infty < y_2 - \beta$. І оскільки відомо, що коефіцієнти cs_2 менші, ніж β, ми знаємо, що додавання cs_2 недостатньо, щоб викликати будь-які переноси шляхом збільшення будь-якого молодшого коефіцієнта, щоб мати величину принаймні y_2. Таким чином, рівняння. (1.1) є істинним, і підпис перевіряється правильно.

Базовий шаблон на рисунку 1.1 досить неефективний, як є. Найяскравіша (але тривіально виправлена) неефективність полягає в тому, що відкритий ключ складається з матриці $k \cdot l$ поліномів, які можуть мати досить велике представлення. Виправлення полягає в тому, щоб згенерувати A з деякого початкового числа за допомогою SHAKE-128, і це стандартна техніка. Отже, відкритий ключ дорівнює (ρ, t), а його розмір домінує t. Новизна Dilithium порівняно з попередніми схемами (наприклад, [12] і qTESLA [13], які є особливим екземпляром фреймворку [12]) полягає в тому, що в Dilithium також зменшується розмір бітового представлення t на дещо більший коефіцієнт ніж два за рахунок збільшення підпису менш ніж на 100 байт.

Основне спостереження для отримання цього дуже вигідного компромісу полягає в тому, що коли верифікатор обчислює w_1' у рядку 13, старші біти $Az - ct$ не надто залежать від молодших бітів t, оскільки t множиться на дуже низький поліном (поліном з низькою вагою) c. У даній схемі деякі молодші біти t не включені у відкритий ключ, і тому верифікатор не завжди може правильно обчислити старші біти $Az - ct$. Щоб компенсувати це, підписувач включає деякі «підказки» як частину підпису, які, по суті, є переносами, викликаними додаванням добутку c із відсутніми молодшими бітами t. За допомогою цієї підказки верифікатор зможе правильно обчислити w_1'.

Крім того, розробники надають можливість зробити дану схему детермінованою за допомогою стандартної техніки додавання початкового числа до секретного ключа та використання цього початкового числа разом із

повідомленням для створення випадковості y в рядку 07. Результат Kiltz та ін. [14] показав, що чим менше різних підписів бачить зловмисник для тих самих повідомлень, тим сильнішим є скорочення в квантовій випадковій моделі оракула між схемою підпису та основними припущеннями про надійність. Хоча незрозуміло, чи існує вдосконалена квантова атака для рандомізованих сигнатур, ми пропонуємо детерміновану версію як варіант за замовчуванням, за винятком сценаріїв, коли супротивник може встановлювати атаки на бічних каналах, які використовують детермінізм [15, 16] . Повна схема Dilithium також використовує основні оптимізації, такі як попереднє гешування повідомлення M, щоб не повторювати його під час кожної спроби підписання.

Зауваження щодо реалізації

Основною алгебраїчною операцією, що виконується на схемі, є множення матриці A, елементами якої є поліноми від $\mathbb{Z}_q[X]/(X^{256}+1)$, на вектор у таких поліномів. У налаштуванні параметрів 3 рівня NIST, A є матрицею 6×5 і, отже, складається з 30 поліномів. Таким чином, множення Ay містить 30 поліноміальних множень. Як і в багатьох схемах на основі решітки, які базуються на операціях над поліноміальними кільцями, в даній схемі було обрано кільце так, щоб операція множення мала дуже ефективну реалізацію через теоретико-числове перетворення (NTT), яке є просто версією FFT, яка працює над кінцеве поле $\mathbb{Z}_q$, а не над комплексними числами. Щоб увімкнути алгоритм NTT з «повним розщепленням», потрібно вибрати просте число q так, щоб група $\mathbb{Z}_q^*$ мала елемент порядку $2n = 512$ або еквівалентно $q = 1\ (mod\ 512)$. Якщо r є таким елементом, то $X^{256}+1 = (X-r)(X-r^3)\cdots(X-r^{511})$ і таким чином можна еквівалентно представити будь-який поліном $a \in \mathbb{Z}_q[X]/(X^{256}+1)$ у формі CRT (китайська теорема про залишки) як $(a(r), a(r^3), \ldots, a(r^{2n-1}))$. Перевагою цього представлення є те, що добуток двох поліномів є координатним. Тому найдорожчими частинами множення поліномів є перетворення $a \to \hat{a}$ та обернене $\hat{a} \to a$ – це NTT і зворотні NTT операції.

Вищезгадана перевага використання представлення NTT є ще більш помітною через те, що при виконанні множення Ay потрібно обчислити набагато менше NTT, ніж розмірність матриці. Оскільки матриця рівномірно випадкова, її можна згенерувати (і зберегти) вже у формі NTT. Виконання множення цієї матриці на вектор $y \in R_q^5$ включає лише виконання 5 обчислень NTT для перетворення y у представлення NTT, потім виконання поточкових множень для отримання Ay у формі NTT, а потім 6 зворотних множень NTT.

Іншою великою трудомісткою операцією є розширення початкового числа ρ в поліноміальну матрицю A. Матриця A потрібна як для підпису, так і для перевірки, тому хороша реалізація SHAKE-128 важлива для ефективності схеми.

Для оптимізованої AVX2 реалізації NTT у Dilithium використовується підхід до використання цілої арифметики, а не з плаваючою комою. Хоча упаковується лише 4 коефіцієнти в один векторний регістр із 256 біт, що є тією самою щільністю, що також використовується реалізаціями з плаваючою комою, можна підвищити швидкість множення приблизно в 2 рази. Цього прискорення було досягнуто шляхом ретельного планування інструкцій і чергування множень та скорочень під час NTT, щоб частини затримок множення були приховані.

Безпека

Дотримуючись основного підходу з [11], безпеку схеми підпису з рисунку 1.1 можна довести в моделі випадкового оракула (ROM) на основі жорсткості двох проблем. Перша — це стандартна проблема LWE (над поліноміальними кільцями), яка вимагає відрізнити $(A, t := As_1 + s_2)$ від (A, u), де u є рівномірно випадковим. Інша проблема полягає в тому, що в [14] було названо проблемою SelfTargetMSIS, яка є проблемою пошуку вектора $\begin{bmatrix} z \\ c \\ v \end{bmatrix}$ з малими коефіцієнтами та повідомленням (дайджестом) μ, що задовольняє

$$\mathrm{H}\left(\mu \parallel [A \,|t|\, \mathrm{I}] \cdot \begin{bmatrix} z \\ c \\ v \end{bmatrix}\right) = c,$$

де A і t рівномірно випадкові, а I є одиничною матрицею. У ROM (Random Oracle model) можна отримати (не жорстке) скорочення за допомогою леми про розгалуження [17, 18] зі стандартної задачі MSIS пошуку z' з малими коефіцієнтами, які задовольняють $Az' = 0$ для SelfTargetMSIS. Можна дотримуватися цього точного підходу, щоб довести безпечність Dilithium у ROM на основі міцності MLWE та SIS.

У моделі квантового випадкового оракула (QROM), де супротивник може запитувати H у суперпозиції, ситуація дещо інша. У [14] було показано, що Dilithium все ще базується на MLWE та SelfTargetMSIS у QROM, навіть із жорстким скороченням, коли схема детермінована. Але більше не можна безпосередньо використовувати лему про розгалуження (оскільки це тип перемотування), щоб дати квантове скорочення від MSIS до SelfTargetMSIS. Існують вагомі підстави вважати SelfTargetMSIS проблемою і, отже, Dilithium безпечні в QROM. По-перше, не існує природних схем підпису, створених з Σ-протоколів з використанням перетворення Фіата-Шаміра, які були б безпечними в ROM, але не в QROM. Крім того, можна встановити параметри Dilithium (залишаючи структуру схеми незмінною), щоб проблема SelfTargetMSIS стала інформаційно теоретично важкою, таким чином залишаючи цю версію Dilithium безпечною в QROM на основі лише MLWE. Реалізація таких параметрів у [14] призводить до схеми з підписами та відкритими ключами, які у 2 та 5 разів більші відповідно. Хоча розробники Dilithium не вважають це хорошим компромісом, існування такої схеми дає додаткову впевненість у безпеці оптимізованої Dilithium.

Нещодавно дві нові роботи ще більше скоротили розрив між безпекою в ROM і QROM. Робота [19] показала, що якщо основний Σ-протокол руйнується і має особливу надійність, то його перетворення Fiat-Shamir є безпечним підписом у QROM. Особлива надійність Dilithium Σ-протоколу

безпосередньо має на увазі твердість MSIS [11, 20]. Крім того, [19] припускає, що протокол Dilithium руйнується. Робота [21] далі показала, що властивість згортання дійсно має зменшення від MLWE. Зменшення є досить незначним, але воно дає ще більше підтвердження того, що немає нічого принципово небезпечного в конструкції Dilithium або будь-якій природній схемі, створеній через структуру Fiat-Shamir, безпеку якої можна підтвердити в ROM. Тож, з'являється все більше доказів того, що розбіжність між захищеними підписами в ROM і QROM незабаром розглядатиметься так само, як відмінність між захищеними схемами в стандартній моделі та ROM – будуть деякі теоретичні відмінності, але безпека на практиці буде та сама.

1.2 Основні операції

Кільцеві операції

Розробники схеми позначають R і R_q відповідно кільця $\mathbb{Z}[X]/(X^n+1)$ і $\mathbb{Z}_q[X]/(X^n+1)$, для q ціле число. Нижче у роботі значення n завжди буде 256, а q буде простим числом $8380417 = 2^{23} - 2^{13} + 1$. За замовчуванням усі вектори будуть векторами-стовпцями. Для вектора v його транспонування позначається через v^T. Логічний оператор $[statement]$ оцінюється як 1, якщо оператор істинний, і як 0 в іншому випадку.

Модульні скорочення. Для парного (відповідно непарного) позитивного цілого числа визначається $r' = r \bmod \pm \alpha$ як унікальний елемент r' у діапазоні $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (відповідно $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$), такий що $r' \equiv r \bmod \alpha$. Іноді це називається центрованим скороченням за модулем α[2]. Для будь-якого натурального числа α визначається $r' = r \bmod^+ \alpha$ як унікальний елемент r' у діапазоні $0 \leq r' < \alpha$ такий, що $r' \equiv r \bmod \alpha$. Якщо точне представлення неважливе, просто пишеться $r \bmod \alpha$.

Розміри елементів. Для елемента $w \in \mathbb{Z}_q$ пишеться $\|w\|_\infty$, що означає $|w \bmod^{\pm} q|$. Визначимо норми l_∞ і l_2 для $w = w_0 + w_1 X + \ldots + w_{n-1} X^{n-1} \in R$:

$$\|w\|_\infty = \max_i \|w_i\|_\infty, \qquad \|w\| = \sqrt{\|w_0\|_\infty^2 + \ldots + \|w_{n-1}\|_\infty^2}.$$

Аналогічно для $w = (w_1, \ldots, w_k) \in R^k$ визначається

$$\|w\|_\infty = \max_i \|w_i\|_\infty, \qquad \|w\| = \sqrt{\|w_1\|^2 + \ldots + \|w_k\|^2}.$$

Будемо писати S_η для позначення всіх елементів $w \in R$ таких, що $\|w\|_\infty \leq \eta$. Ми будемо писати $\tilde{S}_\eta$, щоб позначити множину $\{w \bmod^{\pm} 2\eta : w \in R\}$. Зауважте, що S_η і $\tilde{S}_\eta$ дуже подібні, за винятком того, що $\tilde{S}_\eta$ не містить поліномів з $-\eta$ коефіцієнтами.

Представлення області NTT

Модуль q вибрано таким чином, що існує 512-й корінь з одиниці r за модулем q. Конкретно, ми завжди працюємо з $r = 1753$. Це означає, що циклотомічний поліном $X^{256} + 1$ розкладається на лінійні множники $X - r^i$ за модулем q з $i = 1, 3, 5, \ldots, 511$. Згідно з китайською теоремою про залишки, наше циклотомічне кільце R_q є ізоморфним добутку кілець $\mathbb{Z}_q[X]/(X - r^i) \cong \mathbb{Z}_q$. У цьому добутку кілець легко множити елементи, оскільки тут множення відбувається поточково. Ізоморфізм

$$a \mapsto \left(a(r), a(r^3), \ldots, a(r^{511})\right) : R_q \to \prod_i \mathbb{Z}_q[X]/(X - r^i)$$

можна швидко обчислити за допомогою швидкого перетворення Фур'є. Оскільки $X^{256} + 1 = X^{256} - r^{256} = (X^{128} - r^{128})(X^{128} + r^{128})$, спочатку можна обчислити карту

$$\mathbb{Z}_q[X]/(X^{256} + 1) \to \mathbb{Z}_q[X]/(X^{128} - r^{128}) \times \mathbb{Z}_q[X]/(X^{128} - r^{128})$$

а потім продовжити окремо з двома скороченими поліномами ступеня менше 128, зауважуючи, що $X^{128} - r^{128} = X^{128} - r^{384}$. Швидке перетворення Фур'є також називається перетворенням теорії чисел (NTT) у цьому випадку, коли основне поле є кінцевим полем. Природні швидкі реалізації NTT не виводять вектори з коефіцієнтами в порядку $a(r), a(r^3), \ldots, a(r^{511})$. Таким чином, визначається представлення NTT

області $\hat{a} = NTT(a) \in \mathbb{Z}_q^{256}$ полінома $a \in R_q$, щоб мати коефіцієнти в порядку, як вихід еталонного NTT. Конкретно,

$$\hat{a} = NTT(a) = (a(r_0), a(-r_0), \dots, a(r_{127}), a(-r_{127}))$$

де $r_i = r^{brv(128+i)}$ з $brv(k)$ бітове звернення 8-бітного числа k. З цим позначенням і через властивість ізоморфізму маємо $ab = NTT^{-1} - (NTT(a)NTT(b))$. Для векторів y і матриць A представлення $\hat{y} = NTT(y)$ і $\hat{A} = NTT(A)$ означають, що кожен поліном y_i і $a_{i,j}$, що містить y і A, знаходиться в представленні області NTT.

Гешування

Дана схема використовує кілька різних алгоритмів, які гешують рядки в $\{0,1\}^*$ на представлення різних форм. Нижче наводяться високо рівневі описи цих функцій і описуються деталі того, як саме вони використовуються у схемі підпису.

Гешування до кулі. Нехай B_τ позначає набір елементів R, які мають коефіцієнти, що дорівнюють -1 або 1, а решта дорівнюють 0. Маємо $|B_\tau| = 2^\tau \cdot \binom{256}{\tau}$. Для схеми підпису Dilithium знадобиться криптографічна геш-функція, яка гешує B_τ у два етапи. Перший крок полягає у використанні другої криптографічної геш-функції, стійкої до попереднього зображення, для відображення $\{0,1\}^*$ у домені $\{0,1\}^{256}$. Другий етап складається з XOF (наприклад, SHAKE-256), що відображає вихід першого етапу на елемент B_τ. Причина цього двоетапного процесу полягає в тому, що вихідні дані першого етапу дозволяють дуже простий, компактний опис елемента в B_τ (за припущенням, що XOF неможливо відрізнити від чисто випадкової функції). Алгоритм, який буде використовуватися для створення випадкового елемента у B_τ, іноді називають «вивернутою навиворіт» версією перетасування Фішера-Єйтса [22], і його опис високого рівня наведено на рисунку 1.2. XOF у другому етапі використовується для створення випадковості, необхідної для цього

алгоритму (на кроках 03 і 04), використовуючи вихідні дані першого етапу як початкове значення.

SampleInBall(ρ)
01 Initialize $\mathbf{c} = c_0 c_1 \dots c_{255} = 00 \dots 0$
02 **for** $i := 256 - \tau$ to 255
03 $\quad j \leftarrow \{0, 1, \dots, i\}$
04 $\quad s \leftarrow \{0, 1\}$
05 $\quad c_i := c_j$
06 $\quad c_j := (-1)^s$
07 **return c**

Рисунок 1.2 – Створення випадкового 256-елементного масиву із $r \pm 1$ та $256 - r$ 0′, використовуючи вхідне початкове значення (і XOF), щоб генерувати випадковість, необхідну в кроках 03 і 04.

Розгортання матриці A. Функція ExpandA відображає рівномірне початкове число $\rho \in \{0,1\}^{256}$ на матрицю $A \in R_q^{k \times l}$ у представленні домену NTT. Матриця A потрібна тільки для множення. Отже, для швидшої реалізації функція розширення ExpandA не виводить $A \in R_q^{k \times l} = (\mathbb{Z}_q[X]/(X^{256}+1))^{k \times l}$. Замість цього вона виводить $\hat{A} \in \mathbb{Z}_q^{256}$, що інтерпретується як представлення домену NTT для A. Оскільки A потребує рівномірної вибірки, а NTT є ізоморфізмом, ExpandA також потребує рівномірної вибірки в цьому представленні. Щоб бути сумісною з Dilithium, реалізація, NTT якої створює вектори, упорядковані інакше, ніж еталонний NTT, потребує вибірки коефіцієнтів у непослідовному порядку.

Вибірка векторів y. Функція ExpandMask, яка використовується для детермінованої генерації випадковості схеми підпису, відображає початкове число ρ' і одноразовий κ до $y \in \tilde{S}_{\gamma_1}^l$.

Стійкий до колізій геш. Функція CRH, яка використовується у схемі підпису, є стійкою до колізій геш-функції, що відображається на $\{0,1\}^{384}$.

Біти та підказки вищого/нижчого порядку

Щоб зменшити розмір відкритого ключа, необхідно кілька простих алгоритмів, які витягають «вищий» і «нижчий» біти елементів у $\mathbb{Z}_q$. Мета полягає в тому, щоб, коли задано довільний елемент $r \in \mathbb{Z}_q$ і інший невеликий елемент $z \in \mathbb{Z}_q$, розробникам хотілося б мати можливість відновлювати біти вищого порядку $r + z$ без необхідності зберігати z. Тому визначаються алгоритми, які беруть r, z і створюють 1-бітну підказку h, яка дозволяє обчислити біти вищого порядку $r + z$, використовуючи лише r і h. Ця підказка, по суті, є «перенесенням», викликаним z у додаванні.

Є два різні способи, за допомогою яких розбиваються елементи в $\mathbb{Z}_q$ на їхні «старші» біти та «молодші» біти. Перший алгоритм, $Power2Round_q$, є простим побітовим способом розбиття елемента $r = r_1 \cdot 2^d + r_0$, де $r_0 = r \; mod^{\pm} 2^{\mathrm{d}}$ і $r_1 = (r - r_0)/2^d$.

Слід зауважити, що якщо вибираються представники r_1 як цілі невід'ємні числа від 0 до $[q/2^d]$, тоді відстань (за модулем q) між будь-якими двома $r_1 \cdot 2^d$ та $r_1' \cdot 2^d$ зазвичай дорівнює $\geq 2^d$, за винятком граничного випадку. Зокрема, відстань за модулем q між $[q/2^d] \cdot 2^d$ і 0 може бути дуже малою. Це проблематично у випадку, якщо треба створити 1-бітну підказку, оскільки додавання невеликого числа до r може призвести до зміни старших бітів r більш ніж на 1.

Уникається зміна старших бітів більш ніж на 1 за допомогою простого налаштування. Виберемо α, що є дільником $q - 1$, і запишемо $r = r_1 \cdot \alpha + r_0$ так само, як і раніше. Для простоти припускається, що це парне значення (що можливо, оскільки q непарне). Можливі $r_1 \cdot \alpha$ тепер $\{0, \alpha, 2\alpha, \ldots, q - 1\}$. Слід зауважити, що відстань між $q - 1$ та 0 дорівнює 1, тому $q - 1$ видаляється із набору можливих $r_1 \cdot \alpha$ і просто округлюються відповідні r до 0. Оскільки $q - 1$ і 0 відрізняються на 1, усе це, можливо, збільшує величину залишку r_0 на 1. Ця процедура називається $Decompose_q$. Використовуючи цю процедуру як підпрограму, можна визначити підпрограми $MakeHint_q$ і $UseHint_{\mathrm{q}}$, які

створюють підказку та, відповідно, використовують підказку для відновлення старших бітів суми. Для зручності нотації також визначаються підпрограми $HighBits_{\mathrm{q}}$ і $LowBits_{\mathrm{q}}$, які просто витягують r_1 і r_0 відповідно з результату $Decompose_{\mathrm{q}}$.

Наведені нижче леми визначають властивості цих допоміжних алгоритмів, які необхідні для коректності та безпеки схеми, що розглядається.

Лема 1. Нехай q і α є натуральними числами, що задовольняють $q > 2\alpha$, $q \equiv 1\ (mod\ \alpha)$ і навіть α. Нехай r і z — вектори елементів у R_q, де $\|z\|_\infty \leq \alpha/2$, і нехай h, h' це вектори бітів. Тоді алгоритми $HighBits_{\mathrm{q}}$, $MakeHint_{\mathrm{q}}$ і $UseHint_{\mathrm{q}}$ задовольняють такі властивості:

1) $UseHint_q(MakeHint_q(z, r, \alpha), r, \alpha) = HighBits_q(r + z, \alpha)$.

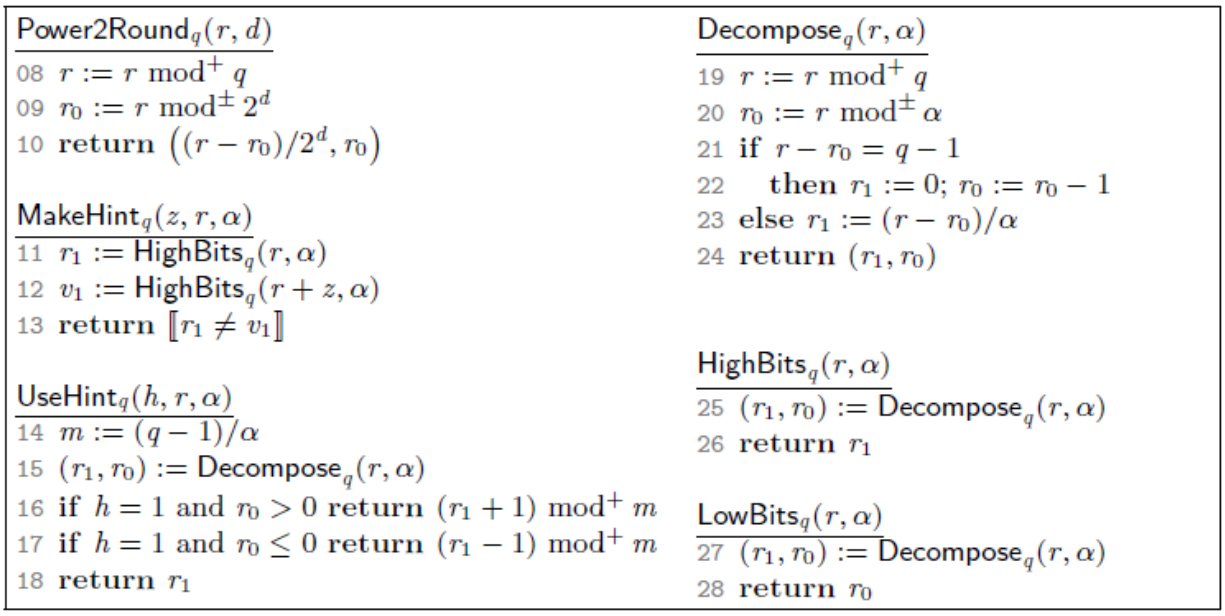

Рисунок 1.3 – Підтримка алгоритмів для Dilithium

2) Нехай $v_1 = UseHint_q(h, r, \alpha)$. Тоді $\|r - v_1 \cdot \alpha\|_\infty \leq \alpha + 1$. Крім того, якщо кількість з 1 у h є ω, тоді всі, крім не більше ω коефіцієнтііів $r - v_1 \cdot \alpha$ матимуть величину не більше $\alpha/2$ після центрованого скорочення за модулем q.

3) Для будь-яких h, h', якщо $UseHint_q(h, r, \alpha) = UseHint_q(h', r, \alpha)$, тоді $h = h'$.

Лема 2. Якщо $\|s\|_\infty \leq \beta$ та $\left\|LowBits_{\mathrm{q}}(r, \alpha)\right\|_\infty < \alpha/2 - \beta$, то

$$HighBits_{\mathrm{q}}(r, \alpha) = HighBits_{\mathrm{q}}(r + s, \alpha)$$

Лема 3. Нехай $(r_1, r_0) = Decompose_q(r, \alpha), (w_1, w_0) = Decompose_q(r + s, \alpha)$, і $\|s\|_\infty \leq \beta$. Потім

$$\|s + r_0\|_\infty < \frac{\alpha}{2} - \beta \Leftrightarrow w_1 = r_1 \wedge \|w_0\|_\infty < \frac{\alpha}{2} - \beta.$$

1.3 Підпис та основні процеси

Алгоритми генерації, підписання та перевірки ключів для схеми підпису Dilithium представлені на рисунку 1.4. Розробники Dilithium представляють детерміновану версію схеми, в якій випадковість, що використовується в процедурі підписання, генерується (з використанням SHAKE-256) як детермінована функція повідомлення та маленький секретний ключ. Оскільки процедуру підписання може знадобитися повторити кілька разів, доки не буде створено підпис, також додається лічильник, щоб вихід SHAKE-256 відрізнявся при кожній спробі підписання того самого повідомлення. Крім того, у зв'язку з тим, що кожне (можливо, довге) повідомлення може потребувати кількох ітерацій для підписання, обчислюється початковий дайджест повідомлення за допомогою хеш-функції, стійкої до колізій, і використовується цей дайджест замість повідомлення під час процедури підписання.

Як обговорювалося вище у розділі, основне вдосконалення дизайну Dilithium порівняно зі схемою на рисунку 1.1 полягає в тому, що розмір відкритого ключа зменшується приблизно вдвічі за рахунок менше сотні додаткових байтів у підписі. Щоб виконати зменшення розміру, алгоритм генерації ключа виводить $t_1 := Power2Round_q(t, d)$ як відкритий ключ замість t, як на рисунку 1.1. Це означає, що замість $\lceil log\ q \rceil$ біт на коефіцієнт,

відкритий ключ потребує $\lceil log\ q \rceil - d$ бітів. У прикладах $q \approx 2^{23}$ і $d = 13$, що означає, що замість 23 бітів у кожному коефіцієнті відкритого ключа є 10.

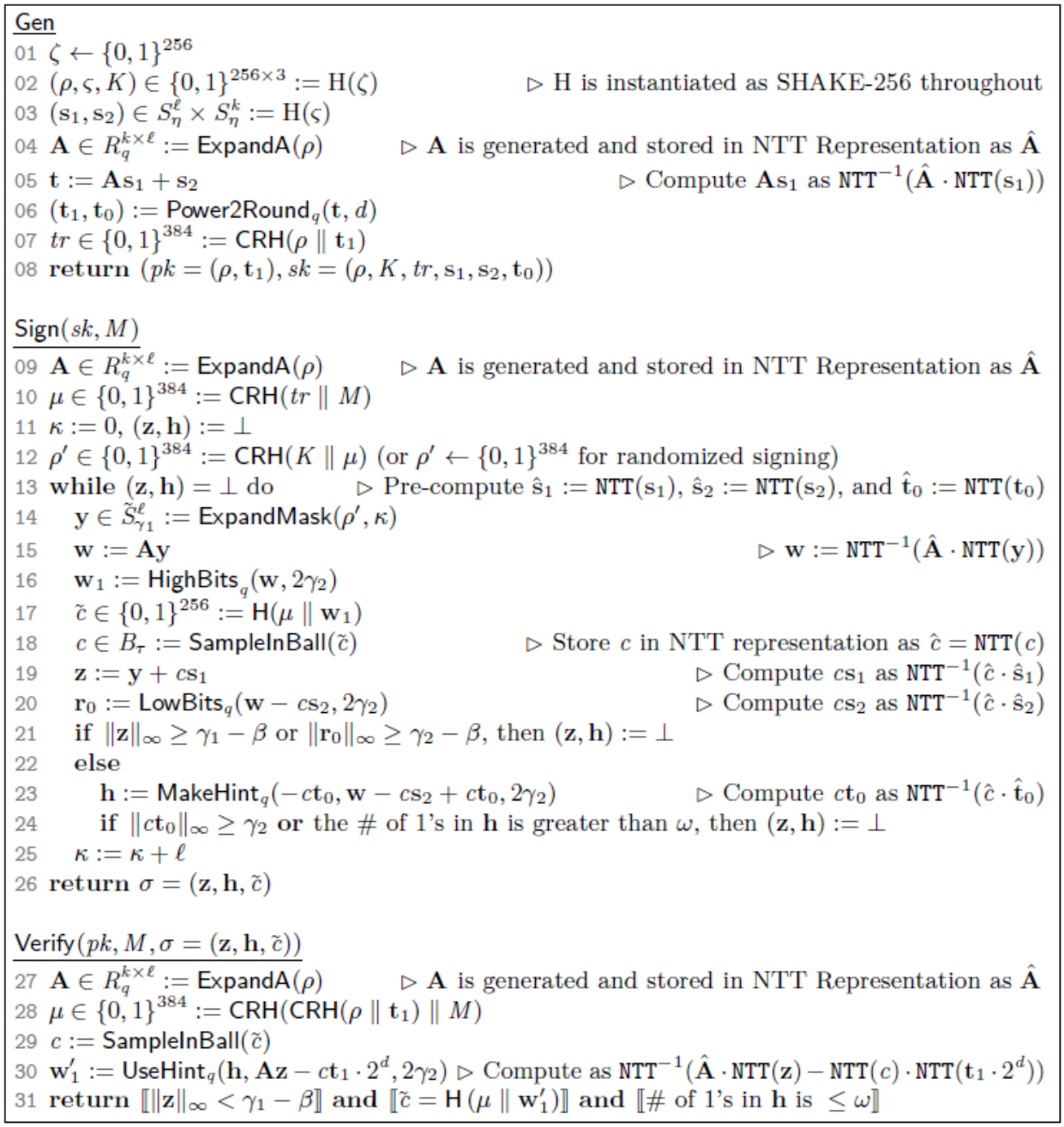

Рисунок 1.4 – Псевдокод для детермінованої та рандомізованої версій Dilithium

Єдина відмінність між двома версіями наведеними на рисунку 1.4 полягає в рядку 12, де ρ' є або функцією ключа та повідомлення, або вибрано абсолютно випадково.

Основна складність, пов'язана з відсутністю цілого t у відкритому ключі, полягає в тому, що алгоритм верифікації більше не може точно обчислити w_1' у рядку 13 на рисунку 1.4. Щоб зробити це, алгоритм верифікації потребуватиме високого порядку бітів $Az - ct$, але він може лише обчислити $Az - ct_1 \cdot 2^d = Az - ct + ct_0$. Незважаючи на те, що старші біти ct_0 дорівнюють 0, його присутність у сумі створює «перенесення», які можуть впливати на старші біти. Таким чином, підписувач надсилає ці переноси як підказку верифікатору. Евристично, виходячи з вибору параметрів, не повинно бути більше ніж ω позиції, в яких виникає перенесення. Таким чином, підписувач просто надсилає позиції, в яких відбуваються ці переноси (це додаткові байти в Dilithium порівняно з підписом на рисунку 1.1), що дозволяє верифікатору обчислити старші біти $Az - ct$.

Примітки щодо впровадження та компроміси щодо ефективності

Щоб зберегти розмір публічного (і секретного) ключа невеликим, обидві процедури $Sign$ і $Verify$ починаються з вилучення матриці A (точніше, її представлення $\widehat{\mathrm{A}}$ домену NTT) із початкового числа ρ. Якщо простір для зберігання не є фактором, тоді $\widehat{\mathrm{A}}$ може бути попередньо обчислено та бути частиною секретного/відкритого ключа. Підписувач може додатково попередньо обчислити представлення домену NTT для s_1, s_2, t_0, щоб трохи прискорити операцію підписання. З іншого боку, якщо підписувач хоче зберегти якомога менший секретний ключ, йому потрібно лише зберегти 32-байтове секретне початкове число, яке використовується для генерації випадковості, для створення ρ, K і s_1, s_2 у алгоритмі генерації ключів. Крім того, можна також підтримувати низький обсяг пам'яті для проміжних обчислень, зберігаючи лише ті частини представлення домену NTT, з якими ви зараз працюєте.

Детерміновані та рандомізовані підписи

Dilithium дає можливість створювати або детерміновані (тобто підпис певного повідомлення завжди однаковий), або рандомізовані підписи. Єдина

відмінність у реалізації між двома параметрами відбувається в рядку 12, де початкове значення ρ' або виводиться з повідомлення та ключа (у детерміністичному підписанні), або вибирається повністю випадковим чином (для рандомізованого підпису).

Можна розглянути рандомізовані сигнатури в ситуаціях, коли застосовуються атаки на бічні канали [15, 16], що використовують детермінізм. Ще одна ситуація, коли можна уникнути детермінізму, це коли підписувач не бажає розкривати повідомлення, яке підписується. Хоча немає витоку часу секретного ключа, є витік часу повідомлення, якщо схема детермінована. Оскільки випадковість схеми випливає з повідомлення, кількість скасувань для певного повідомлення завжди буде однаковою.

Коректність

Далі буде доведена правильність схеми підпису.

Якщо $\|ct_0\|_\infty < \gamma_2$, то за лемою 1 відомо, що

$$UseBits_{\mathrm{q}}(h, w - cs_2 + ct_0, 2\gamma_2) = HighBits_{\mathrm{q}}(w - cs_2, 2\gamma_2).$$

Оскільки $w = Ay$ і $t = As_1 + s_2$, маємо що

$$w - cs_2 = Ay - cs_2 = A\,(z - cs_1) - cs_2 = Az - ct \qquad (1.2)$$

та $w - cs_2 + ct_0 = Az - ct_1 \cdot 2^d$. Тому верифікатор обчислює

$$UseHint_{\mathrm{q}}(h, Az - ct_1 \cdot 2^d, 2\gamma_2) = HighBits_{\mathrm{q}}(w - cs_2, 2\gamma_2).$$

Таблиця 1.1 – Параметри Dilithium

Рівень безпеки NIST	2	3	5
Параметри			
q [модуль]	8380417	8380417	8380417
d [вилучено біти з t]	13	13	13
τ [# з ± 1 у c]	39	49	60

Продовження таблиці 1.1 – Параметри Dilithium

виклик ентропії $[log\binom{256}{\tau} + \tau]$	192	225	257
γ_1 [діапазон коефіцієнтів y]	2^{17}	2^{19}	2^{19}
γ_2 [молодший діапазон округлення]	$(q-1)/88$	$(q-1)/32$	$(q-1)/32$
(k, l) [розміри A]	(4, 4)	(6, 5)	(8, 7)
η [діапазон секретних ключів]	2	4	2
β $[r \cdot \eta]$	78	196	120
ω [макс. # з 1 у підказці h]	80	55	75
Повторення	4.25	5.1	3.85

Крім того, оскільки β встановлено так, що $\|cs_2\|_\infty \leq \beta$ і підписувач перевіряє у рядку 21, що $LowBits_q(w - cs_2, 2\gamma_2) < \gamma_2 - \beta$, лема 2 означає, що

$$HighBits_{\mathrm{q}}(w - cs_2, 2\gamma_2) = HighBits_{\mathrm{q}}(w - cs_2 + cs_2, 2\gamma_2)$$
$$= HighBits_{\mathrm{q}}(w, 2\gamma_2) = w_1. \quad (1.3)$$

Тому w_1' обчислене верифікатором у вхідних даних геш-функції, таке саме, як w_1 підписувача. І таким чином процедура перевірки завжди прийматиметься.

Кількість повторів

Необхідно обчислити ймовірність того, що крок 21 встановить (z, h) значення $\perp$. Імовірність того, що $\|z\|_\infty < \gamma_1 - \beta$, можна обчислити, розглядаючи кожен коефіцієнт окремо. Для кожного коефіцієнта σ з cs_1 відповідний коефіцієнт z буде між $-\gamma_1 + \beta + 1$ та $-\gamma_1 - \beta - 1$ (включно), якщо відповідний коефіцієнт y_i знаходиться між $-\gamma_1 + \beta - 1 - \sigma$ та $\gamma_1 - \beta -$

$1-\sigma$. Розмір цього діапазону дорівнює $2(\gamma_1-1)-1$, а коефіцієнти при y мають $2\gamma_1-1$ варіантів. Таким чином, ймовірність того, що кожен коефіцієнт у знаходиться в гарному діапазоні

$$\left(\frac{2(\gamma_1-\beta)-1}{2\gamma_1-1}\right)^{256\cdot l}=\left(1-\frac{\beta}{\gamma_1-1/2}\right)^{ln}\approx e^{-256\cdot\beta l/\gamma_1}, \qquad (1.4)$$

де використовується той факт, що значення γ_1 є великим порівняно з 1/2.

Далі перейдемо до обчислення ймовірності, яку ми маємо

$$\|r_0\|_\infty=\left\|LowBits_q(w-cs_2,2\gamma_2)\right\|_\infty<\gamma_2-\beta.$$

Якщо (евристично) припустити, що молодші біти рівномірно розподілені за модулем $2\gamma_2$, тоді існує

$$\left(\frac{2(\gamma_2-\beta)-1}{2\gamma_2}\right)^{256\cdot k}\approx e^{-256\cdot\beta k/\gamma_2}$$

ймовірність того, що всі коефіцієнти знаходяться в хорошому діапазоні (використовуючи той факт, що наші значення великі порівняно з 1/2). Таким чином, ймовірність того, що Крок 21 пройде

$$\approx e^{-256\cdot\beta(l/\gamma_1+k/\gamma_2)}. \qquad (1.5)$$

Важче формально обчислити ймовірність того, що крок 24 призведе до перезапуску. Параметри були встановлені так, що евристично $(z,h)=\perp$ з імовірністю від 1 до 2%. Тому переважна більшість повторень циклу буде спричинена кроком 21.

Хотілося б підкреслити, що очікувана кількість ітерацій не залежить від секретного ключа s_1, s_2, і тому жодна інформація про них не може бути отримана за допомогою атаки, яка підраховує ітерації.

1.4 Виклик Dilithium та розширені параметри безпеки

Цифри в дужках для захисту SIS стосуються версії схеми підпису, яка сильно не піддається підробці (тобто важко придумати другий чіткий підпис для вже переглянутої пари повідомлення/підпис).

На додаток до параметрів, наведених у таблиці 1.1 і таблиці Б.1, які відповідають різним рівням безпеки NIST, також надаються параметри, які нижче та вище цих рівнів. Метою двох наборів параметрів нижче рівня 1 NIST (так звані 1-- і 1-) є заохочення криптоаналізу та раннє попередження, якщо проблеми з SIS і LWE виникнуть легше, ніж очікувалося. Рівень 1- - має менше 60 біт безпеки Core-SVP, і тому можна припустити, що нетривіальні обчислювальні зусилля можуть порушити ці параметри в найближчі кілька років. Це також може бути хорошим показником прогресу в обчислювальних проблемах, що лежать в основі Dilithium. Рівень безпеки 1, з іншого боку, має приблизно 90 біт безпеки Core-SVP і розмір блоку BKZ близько 300. Порушення SVP або LWE на цьому рівні в найближчому майбутньому без гігантських обчислювальних зусиль означатиме, що було зроблено великий криптоаналітичний стрибок, тому слід вважати, що запропоновані параметри рівня 2 NIST знаходяться під загрозою. У цьому випадку наполегливо рекомендується негайний перехід на рівні 3 або 5.

На іншому кінці спектру також пропонуються параметри, що перевищують рівень 5 NIST, які розробники називають 5+ і 5++. Для цих наборів параметрів не підвищується надійність будь-яких симетричних примітивів (наприклад, PRF або геш-функцій, стійких до колізій), які використовуються в даному протоколі, а лише підвищується безпека базових проблем решітки. Ціль цих наборів параметрів полягає в тому, щоб показати, якими будуть параметри, якщо буде помірне покращення ґратчастого криптоаналізу. З точки зору стійкості Core-SVP, рівень 5++ має приблизно вдвічі більшу необхідну безпеку, ніж NIST рівень 3, і, отже, добре демонструє, як можуть виглядати параметри, якщо помірні вдосконалення криптоаналізу на основі решітки вимагають змін. Така зміна буде рекомендована у випадку,

якщо якісь майбутні криптоаналітичні зусилля почнуть загрожувати безпеці набору параметрів рівня 3.

1.5 Особливості реалізації

Альтернативний спосіб декомпозиції та обчислення підказок

У псевдокоді для підпису на рисунку 1.4 функція декомпозиції Decompose викликається 3 рази за ітерацію циклу вибірки відхилення. Розробники представили алгоритм таким чином для простоти викладу та для сумісності з доказами безпеки в [14, 20]. У реалізації використовується альтернативний і швидший спосіб обчислення вектора підказки h і умови відхилення на основі молодшої частини w_0 від w. Спочатку зауважте, що лема 3 говорить, що замість обчислення $(r_1, r_0) = Decompose_q(w - cs_2, \alpha)$ і перевірки, чи $\|r_0\|_\infty < \gamma_2 - \beta$ і $r_1 = w_1$, еквівалентно просто перевірити, що $\|w_0 - cs_2\|_\infty < \gamma_2 - \beta$, де w_0 — нижня частина w. Якщо ця перевірка пройдена, $w_0 - cs_2$ є нижчою частиною $w - cs_2$. Далі нагадаємо, що згідно з визначенням функції MakeHint, коефіцієнт полінома в h, обчислений на рисунку 1.4, відмінний від нуля, якщо старші частини відповідних коефіцієнтів $w - cs_2$ і $w - cs_2 + ct_0$ відрізняються. Тепер вже обчислений повний розклад w, де $w = \alpha w_1 + w_0$, і зрозуміло, що $\alpha w_1 + (w_0 - cs_2)$ є правильним розкладом $w - cs_2$. Але тоді $\alpha w_1 + (w_0 - cs_2 + ct_0)$ є правильним розкладанням $w - cs_2 + ct_0$ (тобто старша частина є w_1) тоді і тільки тоді, коли кожен коефіцієнт $w_0 - cs_2 + ct_0$ лежить в інтервалі $(-\gamma_2, \gamma_2]$, або, коли деякий коефіцієнт дорівнює $-\gamma_2$, а відповідний коефіцієнт w_1 дорівнює нулю. Остання умова пов’язана з граничним випадком у функції Decompose. З іншого боку, якщо ці умови не відповідають дійсності, тоді старші частини повинні відрізнятися, і тоді випливає, що для обчислення вектора підказки h достатньо просто перевірити ці умови на коефіцієнти $w_0 - cs_2 + ct_0$.

Біт-пакування

Опис біт-пакування наведено у додатку А.

Гешування до кулі

Функція H на рисунку 1.4 поглинає конкатенацію μ і w_1 у SHAKE-256 і створює 32-байтовий вихід $\tilde{c}$. Тепер точно вкажемо роботу функції SampleInBall : $\tilde{c} \mapsto c \in B_\tau$, описаною на рисунку 1.2, та як вона використовується в схемі підпису. Процедура SampleInBall поглинає 32 байти $\tilde{c}$ у SHAKE-256. Протягом своїх операцій функція стискає SHAKE-256, щоб отримати потік випадкових байтів змінної довжини. Перші біти τ в перших 8 байтах цього випадкового потоку інтерпретуються як випадкові знакові біти $s_i \in \{0,1\}, i = 0, \ldots, \tau - 1$. Решта $64 - \tau$ біти відкидаються. У кожній ітерації циклу for в алгоритмі на рисунку 1.2 використовується вибірка відхилень для елементів з $\{0, \ldots, 255\}$, поки не отримаємо $j \in \{0, \ldots, i\}$. Елемент у $\{0, \ldots, 255\}$ отримується шляхом інтерпретації наступного байта випадкового потоку з SHAKE-256 як числа в цьому наборі. Для підпису s використовується відповідний s_{i-196}.

Розгортання матриці A

Функція ExpandA відображає рівномірне початкове число $\rho \in \{0,1\}^{256}$ на матрицю $A \in R_q^{k \times l}$ у представленні домену NTT. Вона обчислює кожен коефіцієнт $\hat{a}_{i,j} \in R_q$ з $\hat{A}$ окремо. Для коефіцієнта $\hat{a}_{i,j}$ вона поглинає 32 байти з ρ, за якими безпосередньо слідують два байти, що представляють $0 \leq 256 * i + j < 2^{16}$ у порядку байтів від молодшого до байтів у SHAKE-128. У варіанті AES ρ використовується як ключ, а $256i + j$ доповнено нулем до 12-байтового одноразового номера для AES у режимі лічильника. Вихідний потік SHAKE-128 або AES256ctr інтерпретується як послідовність цілих чисел від 0 до $2^{23} - 1$. Це робиться шляхом встановлення старшого біта кожного третього байта на нуль та інтерпретації блоків із 3 послідовних байтів у порядку байтів. Так, наприклад, три байти b_0, b_1 і b_2 використовуються для отримання цілого числа $0 \leq b_2' \cdot 2^{16} + b_1 \cdot 2^8 + b_0 \leq 2^{23} - 1$, де b_2' є логічним I (AND) для b_2 і $2^{128} - 1$. Нарешті, ExpandA виконує відхилену вибірку цих 23-бітних цілих чисел для

вибірки 256 коефіцієнтів $a_{i,j}(r_0), a_{i,j}(-r_0), \ldots, a_{i,j}(r_{127})a_{i,j}(-r_{127})$ з $\hat{a}_{i,j}$ рівномірно з множини $\{0, \ldots, q - 1\}$ у порядку представлення домену NTT.

Вибірка векторів y

Функція ExpandMask відображає (ρ', k) на $y \in \tilde{S}_{\gamma_1}^l$, де $\kappa \geq 0$, і працює наступним чином. Вона обчислює кожен із l коефіцієнтів y, які є поліномами від $\tilde{S}_{\gamma_1}$, незалежно. Для i-го полінома, $0 \leq i < l$, вона поглинає 48 байтів ρ', з'єднаних з 2 байтами, що представлені $\kappa + i$, у порядку байтів від молодшого до байтів у SHAKE-256. У варіанті AES перші 32 байти ρ' використовуються як ключ, а $\kappa + i$ доповнюється нулем до 12-байтового nonce для AES у режимі лічильника. Потім вихідні байти використовуються для створення позитивного числа в діапазоні $\{0, \ldots, 2\gamma_1 - 1\}$, а потім цілі числа, що містять y, отримують шляхом віднімання $(\gamma_1 - 1)$ від цього позитивного числа. Оскільки γ_1 є ступенем 2, вибірка відхилення не потрібна. Оскільки γ_1 дорівнює 17 і 19, для різних рівнів безпеки генерування послідовних цілих чисел, що містять y, передбачає отримання 18 або 20 біт за раз із потоку байтів.

Стійкий до колізій геш

Функція CRH на рисунку 1.4 є геш-функцією, стійкою до колізій. Для цього використовується 384 біта виводу SHAKE-256. CRH викликається з 3 різними входами. Спочатку вона викликається за допомогою $\rho \| t_1$. Потім функція поглинає 32 байти з ρ, а потім $k \cdot 256 \cdot 9/8$ байтів бітового представлення t_1 у SHAKE-256 і приймає перші 48 байтів першого вихідного блоку SHAKE-256 як вихідний геш. Другим входом є $tr \| M$. Тут конкатенація гешу tr і рядка повідомлення M поглинається в SHAKE-256, а перші 48 вихідних байтів використовуються як кінцевий геш. Третій вхід K ‖ μ обробляється таким же чином.

Відкритий ключ

Відкритий ключ містить ρ і t_1, зберігається як конкатенація бітових представлень ρ і t_1 у цьому порядку. Тому він має розмір 32 + 320 Кб.

Секретний ключ

Секретний ключ містить ρ, K, tr, s_1, s_2 і t_0 і також зберігається як конкатенація бітового представлення цих величин у заданому порядку. Отже, секретний ключ потребує $32 + 32 + 48 + 32((k + l) \cdot [log(2\eta + 1)] + 13k)$ байтів. Як згадувалося раніше, підписувач також має можливість просто зберегти 32-байтове значення ς, а потім просто повторно вивести всі інші елементи секретного ключа.

Підпис

Рядок байтів підпису є конкатенацією упакованого в біти представлення z і кодувань h і c у цьому порядку. Далі опишемо кодування h, яке потребує $\omega + k$ байт. Разом усі поліноми вектора h мають не більше ω ненульових коефіцієнтів. Досить зберегти розташування цих ненульових коефіцієнтів. Кожен з перших ω байтів байтового рядка, що представляє h, є індексом і наступного ненульового коефіцієнта в його поліномі, тобто $0 \leq i \leq 255$, або нуль, якщо більше немає ненульових коефіцієнтів. Числа байтів ω аж до $\omega + k - 1$ записують k позицій j границь полінома в рядку ω індекси коефіцієнтів, де $0 \leq j \leq \omega$. Завдання $\tilde{c}$ має 32 байти. Отже, для підпису потрібно $32l \cdot (1 + \log \gamma_1) + \omega + k + 32$ байти.

Реалізація постійного часу

Еталонна реалізація не розгалужується залежно від секретних даних і не отримує доступу до ділянок пам’яті, які залежать від секретних даних. Для модульних скорочень, необхідних для арифметики в R_q, ніколи не використовується оператор «%» мови програмування C. Натомість використовується скорочення Монтгомері без кроків корекції та спеціальних процедур зменшення, які є специфічними для модуля q. Для обчислення функцій округлення, було реалізовано алгоритми без розгалуження. З іншого боку, коли безпечно розкрити інформацію, розробники не намагалися зробити код постійним у часі. Це включає в себе обчислення викликів і умов відхилення в алгоритмі підписання. При виконанні вибірки відхилення в коді виявляється, яка з умов стала причиною відхилення, а у випадку перевірки

норми, який коефіцієнт порушує межу. Це безпечно, оскільки ймовірності відхилення для кожного коефіцієнта не залежать від секретних даних. Завдання розкривають інформацію про $\mathrm{CRH}(\mu \,\|\, w_1)$ також у випадку відхиленого y, але це не розкриває жодної інформації про секретний ключ, коли CRH моделюється як випадковий оракул і w_1 має високу мінімальну ентропію.

Еталонна реалізація

Еталонний NTT — це природна ітераційна реалізація для 32-розрядних цілих чисел зі знаком, яка використовує метеликів Кулі-Тьюкі у прямому перетворенні та метеликів Джентльмена-Санде у зворотному перетворенні. Для модульних скорочень після множення з попередньо обчисленим коренем з одиниці використовується алгоритм Монтгомері, як це вже було зроблено раніше, наприклад, [23]. Для того, щоб скорочені значення були правильними представниками, попередньо обчислені корені містять коефіцієнт Монтгомері $2^{32}\, mod\, q$. Також використовується скорочення Монтгомері після поточкового добутку поліномів у представленнях області NTT. Оскільки на даний момент не можна отримати коефіцієнт Монтгомері, ці продукти фактично є залишками Гензеля. Потім використовується той факт, що перетворення NTT є лінійним і відбувається множення на додатковий коефіцієнт Монтгомері після оберненого NTT, коли ділимо коефіцієнт 256.

Реалізації функцій ExpandA і ExpandMask спочатку стискають кількість вихідних блоків SHAKE-256 і SHAKE-128, які забезпечують достатню випадковість з високою ймовірністю. У випадку ExpandA, який відбирає однорідні поліноми і, отже, потребує принаймні $3 \cdot 256 = 768$ випадкових байтів на поліном, 5 блоків із SHAKE-128 по 168 байт кожен необхідні принаймні для одного полінома. Їх достатньо з імовірністю, більшою за $1 - 2^{-132}$. ExpandMask спочатку отримує 5 блоків із SHAKE-256, які мають 136 байт. Це мінімальна кількість блоків, і її достатньо з імовірністю більше $1 - 2^{-81}$.

Як згадувалося вище, еталонна реалізація захищена від атак із визначенням часу. З цієї причини централізовані залишки у функціях округлення, наведених на рисунку 1.3, не обчислюються з розгалуженням. Замість цього використовується наступний добре відомий прийом, щоб обчислити централізований залишок $r' = r \; mod^{\pm} \; \alpha$, де $0 \leq r < q$. Віднімання $\alpha/2 + 1$ від r дає негативний результат тоді і тільки тоді, коли $r \leq \alpha/2$. Отже, арифметичний зсув цього результату вправо на 31 біт дає -1, тобто ціле число з усіма бітами, що дорівнюють 1, якщо $r \leq \alpha/2$ і 0 інакше. Потім логічне I зміщеного значення і α додається до r, $\alpha/2 - 1$ віднімається. Це призводить до $r - \alpha$ якщо $r > \alpha/2$ та r якщо $r \leq \alpha/2$, тобто централізований залишок.

Розробники активно використовують ліниве скорочення в даній реалізації. У NTT взагалі не скорочуються результати додавання і віднімання. Для округлення та перевірки норми важливо відобразити стандартні представники. Це заморожування коефіцієнтів досягається в постійному часі шляхом умовного віднімання q за допомогою іншого екземпляра арифметичного трюку зсуву вправо.

Оптимізована реалізація AVX2

Була представлена також оптимізована реалізація Dilithium для ЦП, що підтримує набір інструкцій AVX2. Оскільки дві операції, що забирають найбільше часу, — це множення поліномів і розширення матриці та векторів, оптимізована реалізація прискорює ці дві операції.

Для поліноміального множення використовується векторизована версія NTT. Цей NTT забезпечує повне множення двох поліномів, включаючи три NTT, і поточкове множення менш ніж за 5000 циклів Haswell і приблизно в 4,5 рази швидше, ніж еталонний код C, скомпільований за допомогою gcc з увімкненою повною оптимізацією для конкретної машини. На відміну від деяких інших реалізацій (наприклад, [23]), розробники не використовують інструкції з плаваючою комою. При використанні інструкцій із плаваючою комою модульні скорочення легко виконуються шляхом множення на число з

плаваючою комою, оберненого до q, і округлення, щоб отримати коефіцієнт, з якого можна обчислити залишок за допомогою іншого множення та віднімання. Замість цього підходу використовується лише цілочисельні інструкції та та сама методологія скорочення Монтгомері, що й у еталонному коді C. У порівнянні з NTT з плаваючою комою в [23], застосованим до простого числа Dilithium $q = 2^{23} - 2^{13} + 1$, дане ціле число NTT приблизно в два рази швидше.

У будь-який час оптимізований AVX2 NTT має 32 цілих коефіцієнта без знаку, по 32 біта кожен, завантажених у 8 векторних регістрів AVX2. Кожен із цих векторних регістрів містить 4 розширених 64-бітних коефіцієнта. Таким чином, після трьох рівнів NTT скорочені поліноми повністю вписуються в ці 8 регістрів, і їх можна перетворити на лінійні множники без подальших завантажень і збереження. На передостанньому та останньому рівні поліноми мають ступінь менший за 4. Це означає, що кожен поліном вписується в один регістр, але лише половину коефіцієнтів потрібно помножити на корені. З цієї причини перемішуються вектори, щоб згрупувати разом коефіцієнти, які потрібно помножити. Інструкції, які використовуються для цього завдання: perm2i128, vpunpcklqdq і vpunpckhqdq. Вектори перемішуються вже на рівнях 4 і 5, що має перевагу дешевших завантажень для постійних коренів одиниці. Множення виконується за допомогою інструкції vpmuldq. Ця інструкція обчислює повний 64-розрядний добуток двох 32-розрядних цілих чисел. Він має затримку 5 циклів як на Haswell, так і на Skylake. На кожному рівні NTT потрібно помножити половину коефіцієнтів. Тому можна виконувати чотири векторні множення та скорочення Монтгомері паралельно. Це приховує деяку затримку інструкцій множення.

Для швидшого розширення матриць і векторів використовується векторизована реалізацію SHAKE, яка працює на 4 паралельних губках і, отже, може поглинати та видавлювати блоки в ці 4 губки одночасно. Для вибірки це означає, що до чотирьох коефіцієнтів можна відбирати одночасно.

Обчислювальна ефективність

У таблиці Б.2 наведені результати згідно проведених експериментів з визначенням часу з еталонною реалізацією на процесорі Skylake. Вони включають кількість циклів ЦП, необхідних для генерації трьох операцій, підписання та перевірки підпису. Ці цифри є середнім значеннями з вибірки у 1000 спроб. Підписання виконано з розміром повідомлення 32 байти. Комп'ютер, який було використано — це ноутбук із процесором Intel Core i7 6600U, що працює на базовій тактовій частоті до 2600 МГц. Код було скомпільовано за допомогою gcc 7.3.0.

Цифри в дужках для безпеки SIS стосуються сильно непідробної (тобто важко придумати другий чіткий підпис для вже переглянутої пари повідомлення/підпису) версії схеми підпису. Через вибірку відхилень існує помітна різниця між медіанним і середнім часом підписання, тому вимірюються обидва.

1.6 Зменшення безпеки

Стандартним поняттям безпеки для цифрових підписів є безпека UF-CMA, яка є безпекою під час атак на обрані повідомлення. У цій моделі безпеки супротивник отримує відкритий ключ і має доступ до оракула підпису для підпису повідомлень за своїм вибором. Мета зловмисника — створити дійсний підпис нового повідомлення. Дещо суворішою вимогою безпеки, яка іноді корисна, є SUF-CMA (Strong Unforgeability under Chosen Message Attacks), яка також дозволяє супротивнику виграти, створюючи інший підпис повідомлення, яке він уже бачив.

Можна показати, що в (класичній) випадковій моделі оракула Dilithium є SUF-CMA безпечним на основі стійкості стандартних задач MLWE та MSIS. Зниження, однак, не жорстке. Крім того, оскільки розробники схеми також дбають про квантових зловмисників, потрібно розглянути безпеку схеми, коли зловмисник може запитувати геш-функцію на суперпозиції вхідних даних (тобто безпека в моделі квантового випадкового оракула – QROM). Оскільки класичне підтвердження безпеки використовує «лему про розгалуження» (яка,

по суті, є перемотуванням), редукція не переходить до квантового налаштування.

Не існує контрприкладів схем, на безпеку яких фактично впливає негерметичність скорочення. Наприклад, такі схеми, як підписи Шнорра [24], підписи GQ тощо, усі встановлюють свої параметри, ігноруючи негерметичність скорочення. Крім того, єдині відомі способи використання додаткової потужності квантових алгоритмів проти схем, безпека яких базується на квантово стійких проблемах за класичної редукції, включають алгоритми «типу Гровера», які покращують вичерпний пошук (хоча було показано, що не може бути «black-box» доказ того, що перетворення Fiat-Shamir безпечно в QROM [25]).

Причина того, що не було жодних атак із використанням негерметичності скорочення, можливо, пов'язана з тим, що існує проміжна проблема, яка точно еквівалентна, навіть за квантових скорочень, безпеці UF-CMA схеми підпису. Ця проблема, по суті, є «згорткою» основної математичної задачі (такої як MSIS або дискретний журнал) із криптографічною геш-функцією H. Здавалося б, що доки немає зв'язку між структурою математичної задачі та H, вирішення цієї проміжної задачі не легше за вирішення математичної задачі.

Нижче будуть наведені припущення, на яких базується безпека SUF-CMA даної схеми. Перші два припущення, MLWE і MSIS, є стандартними проблемами решітки, які є узагальненням LWE, Ring-LWE, SIS і Ring-SIS. Третя проблема, SelfTargetMSIS, — це згадана вище проблема, яка базується на сумісній надійності MSIS і геш-функції H. У класичному ROM існує (не жорстке) скорочення від MSIS до SelfTargetMSIS.

Припущення

Проблема MLWE. Для цілих чисел m, k і розподілу ймовірностей D : $R_q \rightarrow [0,1]$, говориться, що перевага алгоритму A у розв'язанні вирішальної задачі $MLWE_{m,k,D}$ над кільцем R_q є

$$Adv_{m,k,D}^{MLWE} \coloneqq |\Pr[b = 1 | A \leftarrow R_q^{m \times k};\ t \leftarrow R_q^m;\ b \leftarrow A(A,t)] - \Pr[b = 1 \big| A \leftarrow R_q^{m \times k}; s_1 \leftarrow D^k; s_2 \leftarrow D^m; b \leftarrow A(A, As_1 + s_2)]\big|.$$

Проблема MSIS. З алгоритмом A пов'язується функція переваги $Adv_{m,k,\gamma}^{MSIS}$, щоб розв'язати задачу (нормальної форми Ерміта) $MSIS_{m,k,\gamma}$ над кільцем R_q як

$$Adv_{m,k,\gamma}^{MSIS}(A) \coloneqq \Pr\ [0 < \|y\|_\infty \leq \gamma \wedge [\ \mathrm{I}\ |\ \mathrm{A}\ ] \cdot \mathrm{y} = 0 \mid \mathrm{A}\ \leftarrow \mathrm{A}(\mathrm{A})].$$

Проблема SelfTargetMSIS. Припустимо, що $\mathrm{H} : \{0,1\}^* \to B_\tau$ — криптографічна геш-функція. З алгоритмом A пов'язується функція переваги

$$Adv_{m,k,\gamma}^{SelfTargetMSIS}(A) \coloneqq$$

$$\Pr\left[\begin{matrix} 0 \leq \|y\|_\infty \leq \gamma \\ \wedge\ \mathrm{H}(\mu \parallel [\ \mathrm{I}\ |\ \mathrm{A}\ ] \cdot y) = c \end{matrix} \Big| A \leftarrow R_q^{m \times k};\ \left(y \coloneqq \begin{bmatrix} r \\ c \end{bmatrix}, \mu\right) \leftarrow \mathrm{A}^{|\mathrm{H}(\cdot)\rangle}(\mathrm{A})\right].$$

Класична стійкість SelfTargetMSIS. Далі зображується класичне скорочення від MSIS до SelfTargetMSIS. У цьому сценарії зловмисник A є класичним і має лише класичний доступ до функції H, змодельованої як випадковий оракул. Це скорочення є стандартним застосуванням леми про перемотування/розгалуження, тому дається лише ескіз.

Припустимо, що A робить запити $\mathrm{H}(\mu_1 \parallel w_1), \ldots, \mathrm{H}'(\mu_k \parallel w_k)$ і отримує випадково вибрані відповіді $c_1, \ldots, c_k$ і, нарешті, виводить $\mu_i, y = \begin{bmatrix} r \\ c_i \end{bmatrix}$, що задовольняє $\mathrm{H}(\mu_i \parallel [\ \mathrm{I}\ |\ \mathrm{A}\ ] \cdot y) = c_i$ для деякого $1 \leq i \leq k$. Потім скорочення перемотує A до точки, де було зроблено запит $\mathrm{H}(\mu_i \parallel w_i)$ і перепрограмує відповідь на інший випадково вибраний c_i'. Тоді лема про розгалуження стверджує, що успішне A має $a \approx 1/k$ шанс створити підробку на тому ж i. Тобто воно виведе $\mu_i, y' = \begin{bmatrix} r' \\ c_i \end{bmatrix}$, що задовольняє $\mathrm{H}(\mu_i \parallel [\ \mathrm{I}\ |\ \mathrm{A}\ ] \cdot y') = c_i'$. Отже,

маємо, що [I | A] $\cdot (y - y') = 0$ та $y - y' \neq 0$, оскільки $c_i \neq c_i'$. Крім того, усі коефіцієнти $y_i - y_i'$ малі, що дає рішення для MSIS.

Безпека схеми підпису

Конкретна безпека Dilithium була проаналізована в [14], де було показано, що якщо H є квантовим випадковим оракулом (тобто квантово доступною ідеальною геш-функцією), перевага супротивника A, який порушує безпеку SUF-CMA схема підпису є

$$Adv_{Dilithium}^{SUF-CMA}(A) \leq Adv_{k,l,D}^{MLWE}(B) + Adv_{H,k,l+1,\varsigma}^{SelfTargetMSIS}(C) + Adv_{k,l,\varsigma'}^{MSIS}(D) + 2^{-254} \quad (1.6)$$

для D рівномірний розподіл по S_η , і

$$\varsigma = \max\{\gamma_1 - \beta, 2\gamma_2 + 1 + 2^{d-1} \cdot \tau\}, \quad (1.7)$$

$$\varsigma' = \max\{2(\gamma_1 - \beta), 4\gamma_2 + 2\}. \quad (1.8)$$

Крім того, якщо час виконання та ймовірність успіху (тобто переваги) A, B, C, D є $t_A, t_B, t_C, t_D, \in_A, \in_B, \in_C, \in_D$, то нижня межа $t_A/\in_A$ знаходиться в межах малого коефіцієнта множення $min\ t_i/\in_i$ для $i \in \{B, C, D\}$.

Інтуїтивно зрозуміло, що припущення MLWE потрібне для захисту від відновлення ключа, SelfTargetMSIS є припущенням, на якому базується підробка нових повідомлень, а припущення MSIS необхідне для сильної непідробності. Тепер зобразимо деякі частини підтвердження безпеки, які мають відношення до конкретного налаштування параметра.

Ескіз безпеки UF-CMA. Було показано в [14], що для детермінованих схем підпису з нульовим знанням, якщо супротивник, який має квантовий доступ до H і класичний доступ до оракула підпису, може створити підробку нового повідомлення, тоді також є супротивник, який може створити підробка

без доступу до оракула підпису (тому він отримує лише відкритий ключ). Остання модель безпеки називається UF-NMA – unforgeability under no-message attack. Згідно з припущенням MLWE, відкритий ключ $(A, t = As_1 + s_2)$ не відрізняється від (A, t), де t вибирається однаково навмання. Доказ того, що дана схема підпису – це нульове знання, досить стандартний і відповідає структурі з [9, 11, 12]. Це формально доведено в [14].

Таким чином, якщо припустимо, що проблема $MLWE_{k,l,D}$ є складною, де D є розподілом, який вибирає рівномірне ціле число в діапазоні $[-\eta, \eta]$, тоді, щоб довести безпеку UF-NMA, потрібно лише проаналізувати стійкість експерименту, де супротивник отримує випадковий (A, t), а потім має вивести дійсну пару повідомлення/підпис $M, (z, h, c)$, так що

- $\|z\|_\infty < \gamma_1 - \beta$
- $\mathrm{H}(\mu \parallel UseHint_q(h, Az - ct_1 \cdot 2^d, 2\gamma_2)) = c$
- # з 1 у h є $\leq \omega$

З леми 1 випливає, що можна переписати

$$2\gamma_2 \cdot UseHint_q(h, Az - ct_1 \cdot 2^d, 2\gamma_2) = Az - ct_1 \cdot 2^d + u, \qquad (1.9)$$

де $\|u\|_\infty \leq 2\gamma_2 + 1$. Крім того, лише ω коефіцієнти u матимуть величину більше γ_2. Якщо записати $t = t_1 \cdot 2^d + t_0$, де $\|t_0\|_\infty \leq 2^{d-1}$, тоді можна переписати рівняння (1.9) як

$$\begin{aligned} Az - ct_1 \cdot 2^d + u &= Az - c(t - t_0) + u = Az - ct + (ct_0 + u) \\ &= Az - ct + u' \end{aligned} \qquad (1.10)$$

Зверніть увагу, що найгірша верхня межа для u' є

$$\|u'\|_\infty \leq \|ct_0\|_\infty + \|u\|_\infty \leq \|c\|_1 \cdot \|t_0\|_\infty + \|u\|_\infty \leq \tau \cdot 2^{d-1} + 2\gamma_2 + 1.$$

Таким чином (квантовий) супротивник, якому вдалося створити підробку нового повідомлення, може знайти z, c, u', M такі, що $\|z\|_\infty < \gamma_1 - \beta$, $\|c\|_\infty = 1$, $\|u'\|_\infty \leq \tau \cdot 2^{d-1} + 2\gamma_2 + 1$ і $M \in \{0,1\}^*$ такі, що

$$\mathrm{H}\left(\mu \parallel \frac{1}{2\gamma_2} \cdot [A \mid t \mid \mathrm{I}_k] \cdot \begin{bmatrix} z \\ c \\ u' \end{bmatrix}\right) = c. \tag{1.11}$$

Для простоти запису можемо визначити функцію H' так, що $\mathrm{H}(\mu \parallel x) = \mathrm{H}'(\mu \parallel 2\gamma_2 x)$, і тому рівняння (1.11) стає

$$\mathrm{H}\left(\mu \parallel [A \mid t \mid \mathrm{I}_k] \cdot \begin{bmatrix} z \\ c \\ u' \end{bmatrix}\right) = c. \tag{1.12}$$

Оскільки (A, t) є абсолютно випадковим, це саме те визначення проблеми SelfTargetMSIS, яка була наведена вище.

Стандартний аргумент леми про розгалуження, що наводився вище у роботі, може бути використаний, щоб показати, що супротивник, який розв'язує вищезазначену проблему в (стандартній) моделі випадкового оракула, може бути використаний для вирішення проблеми MSIS. Хоча редукція за допомогою леми про розгалуження є хорошою «перевіркою розумності», вона не особливо корисна для встановлення параметрів через відсутність стійкості. Отже, як встановити параметри? Перетворення Фіата-Шаміра використовувалося більше 3 десятиліть (і ми знаємо про нещільність леми про розгалуження для двох з них), але налаштування параметрів для схем, які його використовують, ігнорували цю втрату тісноти. Таким чином, неявно ці схеми покладаються на точну стійкість аналогів (на основі різних припущень, таких як дискретний логарифм [24], односторонність RSA тощо) проблеми в рівнянні (1.12).

Інтуїція для безпеки проблеми в рівнянні (1.12) (та його аналоги дискретного логарифму, RSA тощо) виглядає наступним чином: оскільки H є криптографічною геш-функцією, структура якої повністю незалежна від алгебраїчної структури її вхідних даних, вибір деякого μ «стратегічно» не

повинен допомогти – отже, проблема була б такою ж важкою, якби μ було виправлено. Потім, знову покладаючись на незалежність H' та алгебраїчну структуру його вхідних даних, єдиним підходом для отримання рішення є вибір деякого w, обчислення $\mathrm{H}'(\mu \parallel w) = c$, а потім знаходження z, u' таких, що $Az + u' = w + ct$. Важкість знаходження таких z, u' з l_∞-нормами менша, ніж у рівнянні (1.7) так що

$$\mathrm{Az} + u' = t' \tag{1.13}$$

для деякого t' є проблемою, конкретну безпеку якої буде проаналізовано. Зауважте, що ця межа є дещо консервативною, оскільки в рівнянні (1.12) $\|z\|_\infty < \gamma_1 - \beta \ll \tau \cdot 2^{d-1} + 2\gamma_2 + 1$. Крім того, лише ω коефіцієнти u' можуть бути більшими, ніж $\tau \cdot 2^{d-1} + \gamma_2$. Таким чином, більшість коефіцієнтів розв'язку (z, u') для рівняння (1.13) створені справжньою підробкою, повинні мати коефіцієнти, помітно менші за максимум у рівнянні (1.7).

Додавання сильної властивості непідробності (Strong Unforgeabilty Property). Щоб виконати вимогу сильної непідробності, потрібно обробити додатковий випадок. Інтуїтивно зрозуміло, що скорочення від UF-CMA до UF NMA використало той факт, що підробка нового повідомлення обов'язково вимагатиме використання виклику c, для якого зловмисник ніколи не бачив дійсного підпису (тобто (z, h, c) ніколи не було виходом підписуючого оракула). Щоб довести сильну непідробність, також слід розглянути випадок, коли супротивник бачить підпис (z, h, c) для μ, а потім лише змінює (z, h). Іншими словами, опонент отримує два дійсні підписи, такі що

$$UseHint_q(h, Az - ct_1 \cdot 2^d, 2\gamma_2) = UseHint_q(h', Az' - ct_1 \cdot 2^d, 2\gamma_2).$$

За лемою 1 можна показати, що наведена рівність означає, що існують $\|z\|_\infty \leq 2(\gamma_1 - \beta)$ і $\|u\|_\infty \leq 4\gamma_2 + 2$ такі, що $Az + u = 0$.

Конкретний аналіз безпеки

Розробниками схеми було проведено аналіз найвідоміших ґратчастих атак проти проблем у рівнянні (1.6), на якому базується безпека даної схеми підпису. Найкращі атаки включають пошук коротких векторів у деякій решітці. Основна відмінність між проблемами MLWE і MSIS полягає в тому, що проблема MLWE передбачає пошук короткого вектора в решітці, в якій існує «незвичайно короткий» вектор. Проблема MSIS, з іншого боку, передбачає просто знаходження короткого вектора у випадковій решітці. У термінології рюкзака проблема MLWE — це ранець низької щільності, тоді як MSIS — екземпляр рюкзака високої щільності. Аналіз цих двох випадків дещо відрізняється, і розробники аналізують проблему MLWE і проблему MSIS (а також SelfTargetMSIS) у [26].

Спочатку розробники схеми дотримуються методології твердості ядра SVP з [23], заснованої на асимптотичній вартості просіювання [27, 28], що є значно консервативнішим рішенням ніж інші, що використовуються в криптографії на основі решітки. Однак останній прогрес у розсіюванні на практиці, здається, вимагає більш уточнених оцінок, тож дані уточнення надаються в [26]. Це виражає безпеку з точки зору кількості воріт, необхідних для обчислення, у моделі RAM; оскільки алгоритм [23] забезпечує довільний доступ до експоненціально великого обсягу пам’яті, цілком ймовірно, що будь-яка реалістична реалізація буде значно дорожчою.

Незважаючи на те, що проблеми MLWE та MSIS визначаються над поліноміальними кільцями, наразі не виявлено жодного способу використання цієї кільцевої структури, і тому найкращі атаки встановлюються, просто розглядаючи ці проблеми як проблеми LWE та SIS. Проблеми LWE та SIS точно такі ж, як у визначеннях MLWE та MSIS вище в роботі, де йдеться про заміну кільця R_q на $\mathbb{Z}_q$.

Консервативний вибір для проблеми MSIS

Безпека Dilithium базується на жорсткості проблем типу LWE та SIS. Схеми шифрування на основі решітки базуються виключно на проблемі LWE

та її похідних із доданою структурою (наприклад, MLWE, NTRU тощо), природно, що проблемам типу LWE у майбутньому приділятиметься більше уваги. Хоча обидві проблеми досить схожі в тому, що вони вимагають пошуку коротких векторів у певній решітці, і, крім того, найкращі алгоритми мають однаковий основний компонент, тож розробники схеми вважають, що має сенс бути трохи більш консервативним щодо параметрів на стороні SIS. Розробники вже були досить консервативними, ігноруючи той факт, що лише дуже небагато (тобто ω) коефіцієнтів розв'язку можуть мати максимальну l_∞-довжину, а також той факт, що було втрачено адитивний коефіцієнт $\tau \cdot 2^{d-1}$, оскільки хотіли отримати зменшення від неоднорідної проблеми SIS з метою t, а не t_1, навіть якщо не відомо жодної покращеної атаки проти останньої.

Є ще одне місце, де розробники схеми могли б бути ще більш консервативними проти атак, про існування яких невідомо. Класичне скорочення від SelfTargetMSIS до MSIS збільшує «l_∞-норму вилученого рішення в 2 рази. Більш конкретно, довжина вектора $y_i - y_i'$ і вдвічі більша за довжину ζ в рівнянні (1.7). Немає відомих алгоритмів, які б використовували цю перевагу, і розробники схеми не враховували ці втрати зменшення під час обчислення жорсткості проблеми SIS. Заради додаткового консерватизму тепер надається стійкість SIS, припускаючи, що справді є певна можлива користь від втрати цього фактора. Замість класичної безпеки Core-SVP SIS 124, 186 і 265 для рівнів 2, 3 і 5 NIST можна було б мати відповідну безпеку 115, 172 і 256. Таким чином, навіть у цьому ультраконсервативному випадку все ще достатньо безпеки SIS у представлених наборах параметрів.

Зміна рівнів безпеки

Найпростіший спосіб підвищити/знизити безпеку Dilithium — це змінити значення (k, l), а потім відповідно адаптувати значення η (а потім β і ω). Збільшення (k, l) на 1 кожен призводить до збільшення відкритого ключа на 300 байт і підпису на 700 байт; і підвищує безпеку на 30 біт.

Іншим способом підвищення безпеки було б зниження значень γ_1 і/або γ_2. Це ускладнило б підробку підписів (твердість яких базується на основній проблемі SIS). Замість того, щоб збільшити розмір відкритого ключа/підпису, негативним ефектом зниження γ_i є те, що підпис вимагатиме більше повторень. Можна так само збільшити значення η, щоб зробити проблему LWE складнішою за рахунок більшої кількості повторень. Оскільки збільшення часу виконання є досить різким (наприклад, зменшення вдвічі обох γ_i призвело б до зведення в квадрат кількості необхідних повторень відповідно до рівняння (1.5)), тож рекомендується збільшити (k, l), коли потрібно «суттєво» підвищити безпеку. Зміна γ_i повинна бути зарезервована лише для незначних «коригувань» у рівнях безпеки.

1.7 Висновки за розділом

1) У цьому розділі була розглянута та описана схема Dilithium. Dilithium – схема цифрового підпису, безпека якої базується на складності пошуку коротких векторів у решітках. Детально описувалася її математична модель, принципи підписання, генерації ключів та інші алгоритми та методи на яких заснована ця схема.
2) Безпеку схеми підпису можна довести в моделі випадкового оракула (ROM) на основі жорсткості двох проблем. Перша — це стандартна проблема LWE (над поліноміальними кільцями), яка вимагає відрізнити $(A, t := As_1 + s_2)$ від (A, u), де u є рівномірно випадковим. Інша проблема полягає в тому, що в [14] було названо проблемою SelfTargetMSIS, яка є проблемою пошуку вектора $\begin{bmatrix} z \\ c \\ v \end{bmatrix}$ з малими коефіцієнтами та повідомленням (дайджестом) μ, що задовольняє

$$\mathrm{H}\left(\mu \parallel [A \,|t|\, \mathrm{I}] \cdot \begin{bmatrix} z \\ c \\ v \end{bmatrix}\right) = c.$$

3) Незважаючи на те, що проблеми MLWE та MSIS визначаються над поліноміальними кільцями, наразі не виявлено жодного способу використання цієї кільцевої структури, і тому найкращі атаки встановлюються, просто розглядаючи ці проблеми як проблеми LWE та SIS. Проблеми LWE та SIS точно такі ж, як у визначеннях MLWE та MSIS вище в роботі, де йдеться про заміну кільця R_q на $\mathbb{Z}_q$.
4) Попередній аналіз дозволяє зробити висновок, що сучасні варіанти механізмів LWE ґрунтуються на поліноміальних кільцях, зокрема на $R_q = \mathbf{Z}_q[X]/(x^{2^n}+1)$. Властивості поліномів кільця дозволяють довести ряд теоретичних тверджень щодо стійкості криптосистеми і розробляти ефективні програмні реалізації. Проте, такі кільця мають нетривіальні підполя, що теоретично може використовуватися для криптоаналізу, проте на практиці атак, що застосовують ці додаткові структури, не було знайдено, або ці атаки знаходяться в незавершеному вигляді досліджень.
5) Проблеми криптоаналізу RLWE та MLWE по суті зводяться до LWE проблеми. Таке зведення можливо для $R_q = \mathbf{Z}_q[X]/(x^{2^n}+1)$, оскільки доведено, що RLWE є не менш стійким, ніж LWE. Проте, при такому підході внутрішня структура кільця ігнорується.
6) Безпека Dilithium базується на жорсткості проблем типу LWE та SIS. Схеми шифрування на основі решітки базуються виключно на проблемі LWE та її похідних із доданою структурою (наприклад, MLWE, NTRU тощо), природно, що проблемам типу LWE у майбутньому приділятиметься більше уваги. Хоча обидві проблеми досить схожі в тому, що вони вимагають пошуку коротких векторів у певній решітці, і, крім того, найкращі алгоритми мають однаковий основний компонент, тож розробники схеми вважають, що має сенс бути трохи більш консервативним щодо параметрів на стороні SIS.

7) Атаки на решітках полягають у зведенні проблеми LWE до достатньо вивчених теоретичних проблем в теорії решіток. Існують 3 основні підходи: зведення LWE до BDD, зведення LWE до SIS, зведення LWE до uSVP. Кожен з цих підходів в кінцевому випадку зводиться до задачі пошуку достатньо малого вектора на решітці, для чого використовується алгоритм BKZ та його варіації.
8) Точні оцінки для BKZ та його варіацій не відомі. При практичній оцінці використовується ряд еврістичних підходів та екстраполяція результатів, що отримані на решітках меншої розмірності. Це становить основну проблему при оцінці криптостійкості систем подібних Dilithium, оскільки немає гарантії, що не з'явиться кращого способу редукції решіток, або оцінка виявиться недопустимо не точною.
9) Деталізовані дві атаки називаються Primal Attack і Dual Attack відповідно і складають основу для оцінки складності криптосистем на основі LWE (та його різновидів). Всі інші покращення як правило є еврістиками.
10) Атака підробки ЕП здійснюється шляхом вирішення задачі SIS, причому дуальна атака фактично є вирішенням SIS. Проте автори ЕП Dilithium запропонували інший підхід до вирішення цієї задачі. В SIS потрібно знайти вектор, всі елементи якого менші за певну величину. Автори Dilithium пропонують шукати малі вектори на решітці до тих пір, доки всі елементи такого малого вектора не будуть менші за задану величину в задачі SIS.

2 ЗАГАЛЬНІ ВІДОМОСТІ ТА ПАРАМЕТРИ СТАНДАРТІВ ЕП «ВЕРШИНА 1» ТА «ВЕРШИНА 2»

Ціллю цього розділу є розгляд вітчизняних технологій електронного підпису їх короткий опис, аналіз та розгляд основних властивостей.

Теоретичні дослідження в криптології, зокрема в Україні, направлені на оцінку, порівняльний аналіз та удосконалення існуючих та перспективних асиметричних криптоперетворень типу АСШ, ПІК, ЕП, а також симетричних криптоперетворень типу БСШ та ПСШ.

Наразі розроблені, вже прийняті в якості національних чи впроваджуються в Україні такі стандарти КЗІ:

- БСШ ДСТУ 7624:2014;
- функція гешування ДСТУ 7564:2014;
- ПСШ ДСТУ 8845:2019;
- АСШ КЕМ ДСТУ 8961:2019.

На етапі обговорення та прийняття в якості національних знаходяться проєкти стандартів ЕП «Вершина 1» та «Вершина 2».

Особливістю цих стандартів є те, що в них забезпечується рівень безпеки включно до 512 біт від класичних атак та 256 біт від квантових, а також від атак на основі помилок та спеціальних атак.

Стандарт ДСТУ 8961:2019 розроблено на основі математики алгебраїчних решіток, а проєкти стандартів «Вершина 1» та «Вершина 2» – на основі алгебраїчних решіток з відхиленням та алгебраїчних решіток з використанням спеціальних даних відповідно. Стандарти «Вершина 1» та «Вершина 2» базуються на математичних методах схем електронного підпису Crystals-Dilithium та Falcon відповідно.

Особливістю реалізації вказаних криптоперетворень є оптимізація по швидкодії, що досягається оптимізацією на програмному рівні за рахунок застосування швидких перетворень при множенні поліномів.

2.1 Генерація загальносистемних параметрів ЕП Dilithium («Вершина 1»)

В даному підрозділі описується генерація загальносистемних параметрів для 128, 256, 384, 512 біт стійкості.

Для генерації системних параметрів удосконаленого ЕП Dilithium використовуються отримані розробниками схеми результати аналізу відомих атак на криптосистему [26]. Тож слід встановити умови, при яких буде забезпечено захист від цих атак. Найбільш очевидним вибором атаки на криптосистему є відновлення приватного ключа, використовуючи значення відкритого ключа.

Атака відновлення приватного ключа на основі відкритого ключа

Для схеми електронного підпису Dilithium вирішення цієї задачі зводиться до задачі MLWE. Для аналізу загалом MLWE (Module learning with errors) використовується стратегія зведення до проблеми LWE [26, 29]. Це полегшує безпосередній аналіз, а також залишає простір для оптимізацій. Проблема LWE може бути вирішена різними шляхами.

До першого можливо віднести такі комбінаторні атаки, як зустріч посередині, BKW чи Arora-Ge. Такі атаки ефективні в ситуації, коли коефіцієнти полінома, що є особистим ключем, належать до деякої невеликої множини. Як правило, їх розгляд має сенс тільки у разі гомоморфних криптосистем [26, 29]. Надалі для ЕП Dilithium вони не будуть розглядатися, оскільки значно поступаються в ефективності.

Інший шлях – використання атак через редукцію решіток. Розглядаються атаки на решітці та дуальній до неї [29]. У першому випадку будується решітка (2.1)

$$\Lambda = \{x \in Z^{m+n+1} : (A|I_m| - c)^* x = 0 \; mod \; q \qquad (2.1)$$

Далі за допомогою алгоритмів редукції базису слід знайти найменший унікальний вектор (задача фактично зводиться до USVP) [30]. Для оцінки стійкості до цієї атаки зазвичай використовується підхід, який було запропоновано в роботі [29]. Фактично виконується пошук найменшого розміру блоку β для BKZ, такого, що виконується умова

$$\sigma\sqrt{\beta} \leq \delta_0^{2\beta-d-1} q^{\frac{n}{q}},$$

де σ, q – загальносистемні параметри, d – розмірність решітки, δ_0 – максимальне значення фактора Ерміта, за яким решітка буде достатньо редукованою. Причому δ_0 можливо виразити через β як $\delta \approx ((\pi\beta)^{\frac{1}{\beta}} * \beta/2\pi e)^{1/2(\beta-1)}$. Тож отримуємо нелінійне діофантове рівняння. Дане рівняння можна вирішувати повним перебором, або іншими більш оптимізованими шляхами.

Для дуальної атаки будується решітка типу (2.2)

$$\mathrm{L} = \left\{(x, y) \in Z_q^m \times Z_q^n \mid A^* x = 0 \; mod \; q\right\}. \qquad (2.2)$$

Для виконання цієї атаки, задача полягає у тому, щоб визначити, чи буде знайдений після редукції вектор на цій решітці належати до розподілу, з якого отримано ключ, чи рівномірного розподілу. Згідно з [29] статистична відстань між цими двома розподілами складає $\varepsilon = 4\exp(-2\pi^2 \left(\frac{l\sigma}{q}\right)^2$, де l – довжина найменшого вектора. Відповідно, потрібно перебрати $1/\varepsilon^2$ для вдалого виконання атаки даних векторів. Нехай вирішувач працює за час T, тоді кількість спроб можливо розрахувати як $R = \max(1, \frac{1}{T*\varepsilon^2})$. Базуючись на тому, що довжина вектора l залежить від фактора Ерміта, як $\delta_0^{d-1} q^{\frac{n}{d}}$, а його можна виразити через розмір блока як

$$\delta \approx ((\pi\beta)^{\frac{1}{\beta}} * \beta/2\pi e)^{1/2(\beta-1)},$$

то задача знову зводиться до перебору значень β.

Дві атаки описані вище називають Primal Attack і Dual Attack відповідно. Вони складають основу для оцінки складності різних криптосистем на основі LWE, також його різновидів [29, 30]. Всі інші покращення являються як правило евристиками.

Атаки способом підробки електронного підпису

Однин з шляхів для здійснення атак – підробка ЕП. У цьому разі атака здійснюється шляхом вирішення задачі SIS. Слід взяти до уваги, що дуальна атака фактично є вирішенням SIS [29, 30], але автори Dilithium запропонували інший підхід до вирішення даної задачі. В SIS потрібно знайти вектор, в якому всі елементи будуть менші за певну величину. Розробники Dilithium пропонують шукати малі вектори на решітці до тих пір, доки всі елементи такого малого вектора не будуть менші за задану величину в задачі SIS. Нехай T – час роботи вирішувача, p – ймовірність знаходження такого вектора. Тоді ймовірність атаки можливо оцінити як $\frac{1}{T*p}$. Тож дана ймовірність має зрости до потрібного рівня, якщо обчислення буде повторене декілька разів. Недоліком даного підходу є використання великої кількості евристичних міркувань, але складність оцінки криптостійкості виявляються меншою, ніж при використанні дуальної атаки [31].

Після редукції решітки в векторах умовно можна виділити три зони. У першій зоні довжини норм Грамма-Шмідта будуть розподілені рівномірно за модулем q, у другій довжини норм Грамма-Шмідта матимуть нормальний розподіл, а у третій усі довжини норм Грамма-Шмідта дорівнюють 0. Якщо рандомізувати базис решітки перед редукцією, то перша зона зникає і редукція відбувається швидше [31].

Таким чином, потрібно, щоб перші j довжини норм Грамма-Шмідта (що розподілені за нормальним розподілом) були менші за задане значення ζ. Ймовірність цього складає (2.3)

$$p = \Phi\left(\frac{\zeta}{\sigma\sqrt{2}}\right)^{j+1} \quad (2.3)$$

Головна проблема полягає у знаходженні необхідного значення j. Варто окремо згадати про алгебраїчні атаки. При достатньо великих q та достатньо малих σ існують алгебраїчні атаки на LWE [32]. Для значень σ та q, що використовуються в Dilithium ці атаки не можуть бути здійснені.

Обчислення основних параметрів для удосконаленого ЕП Dilithium

В таблиці Б.3 наводяться основні параметри, що потрібні для реалізації SIS та LWE атак.

Чим менше є значення N, тим швидше працюватиме схема. Розробники ЕП Dilithium використовують $N = 256$. Збільшуючи кількість поліномів через параметри (k, l) можливо досягти будь-якого рівня стійкості до MLWE, проте для забезпечення стійкості до інших атак при використанні $N = 256$ вже для рівня стійкості 384 біт не існує стійких параметрів, тому для $\lambda \in \{384, 512\}$ потрібно збільшити N до 512.

Параметр q сильно впливає на стійкість, а також повинен бути простим числом. Для створення ефективних реалізацій з використанням NTT повинно виконуватися співвідношення $q \equiv 1 \, mod \, 2N$. Розробники Dilithium використовують $q = 8389417$. Немає сенсу збільшувати значення q, тому що при його збільшенні q буде знижуватися криптостійкість, а також тому що умова $q \equiv 1 \, mod \, 2N$ виконується для 256, 384, 512 біт безпеки [31]. Для обчислення параметрів (γ_1, γ_2) розробники використовують формулу (2.4)

$$\gamma_1 = \frac{q-1}{16}, \qquad \gamma_2 = \gamma_1/2 \quad (2.4)$$

Стійкість до всіх можливих атак буде приблизно однаковою при такому виборі. Такий вибір має сенс і при генерації параметрів для 256, 384, 512 біт

безпеки. Параметр h визначає кількість ненульових елементів в поліномі c. Для стійкості λ кількість можливих поліномів повинна бути не меншою за 2^{λ}. Тож, має виконуватися нерівність (2.5)

$$2^h \binom{n}{h} \geq 2^{\lambda}. \tag{2.5}$$

Атаки на LWE можуть бути задані трьома параметрами: n, q, σ, де n – розмірність решітки (у випадку Dilithium $n = N^*l + 1 + m, m \in \{0, N^*k\}$), q – найбільше значення елементів векторів. Співпадає зі значенням q в загальносистемних параметрах. σ – середньоквадратичне відхилення для розподілу, з якого отримується секретний ключ. Для ЕП Dilithium це значення обчислюється за формулою (2.6)

$$\sigma = \sqrt{\frac{\sum_{i=1}^{\eta} i^2}{2\eta + 1}} \tag{2.6}$$

Атаки на SIS параметризовано трьома параметрами: x, y, ς. Для ЕП Dilithium $x = N^*k, y = N^*l$. Параметр ζ – обмеження на максимальне значення елементів векторів. Для ЕП Dilithium є дві атаки, що зводяться до SIS, тому відповідно два різних значення ζ

$$\begin{aligned} \zeta_1 &= \max(\gamma_1 - \beta, 2\gamma_2 + 1 + 2^{d-1}h) \\ \zeta_2 &= \max(2(\gamma_2 - \beta), 4\gamma_2 + 2) \end{aligned} \tag{2.7}$$

Якщо встановити обмеження $2^{d-1}h + 1 \leq 2\gamma_2$ і врахувати що $\gamma_2 = \gamma_1/2$, то маємо:

$$\zeta_1 \leq 4\gamma_2$$

$$\zeta_2 \leq 4\gamma_2 + 2 \tag{2.8}$$

Параметр β визначає максимальне значення коефіцієнтів поліномів $c * s_i$. Враховуючи обмеження вище, максимальне значення яке може бути отримане це $\beta = \eta h$. При такому виборі забезпечуватиметься максимальний захист від атак SIS, проте параметр β при всіх можливих значеннях майже не впливає на безпеку, тож його вплив можливо не враховувати. Проте, параметр β сильно впливає на ймовірність повтору циклу. Ця ймовірність становить

$$\approx \exp(-N\beta\left(\frac{l}{\gamma_1} + \frac{k}{\gamma_2}\right)) \tag{2.9}$$

Тож, саме ця ймовірність є критерієм вибору параметра β.

За допомогою параметру w можна зменшити розмір підпису ціною повтору циклу, а також він не впливає на криптостійкість. Практичні експерименти показали, що при значенні $w = 0.08nk$ отримуються гарні результати, проте можливі подальші оптимізації [31].

Враховуючи приведені критерії, алгоритм генерації загальносистемних параметрів виглядає наступним чином:

1) Визначити потрібний рівень безпеки $\lambda \in \{256, 384, 512\}$;
2) Обрати значення N. Якщо $\lambda = 256$, то $N = 256$, інакше $N = 512$;
3) Обрати значення q. $q = 8389417$;
4) Обчислити γ_1 та γ_2 за формулами $\gamma_1 = (q-1)/16$, $\gamma_2 = \gamma_1/2$;
5) Обчислити значення η. За замовченням встановити $\eta = 2$. На наступних кроках значення буде уточнене;
6) Встановити значення $(k, l) = (2,1)$;
7) Обчислити λ_1-стійкість до Primal Attack. Якщо стійкість менша за λ, то оновити параметри $(k, l) = (k+1, l+1)$ та повернутися до кроку 7 або збільшити η та повернутися до кроку 7;

8) Обчислити λ_2-стійкість до Dual Attack. Якщо стійкість менша за λ, то оновити параметри $(k, l) = (k + 1, l + 1)$ та повернутися до кроку 7 або збільшити η та повернутися до кроку 7;
9) Обчислити λ_3-стійкість до SIS з ζ_1. Якщо стійкість менша за λ, то оновити параметри $(k, l) = (k + 1, l + 1)$ та повернутися до кроку 7 або збільшити $k = k + 1$ та повернутися до кроку 7;
10) Обчислити λ_3-стійкість до SIS з ζ_2. Якщо стійкість менша за λ, то оновити параметри $(k, l) = (k + 1, l + 1)$ та повернутися до кроку 7 або збільшити $k = k + 1$ та повернутися до кроку 7;
11) Обчислити h як найбільше ціле, для якого виконується нерівність $2^h \binom{n}{h} \geq 2^\lambda$;
12) Обчислити d як найбільше ціле, для якого виконується нерівність $2^{d-1}h + 1 \leq 2\gamma_2$;
13) Встановити $\beta = \eta h$ та зменшувати β, щоб ймовірність повтору циклу була достатньо малою;
14) Обчислити $w = 0.08nk$ (цей крок не впливає на криптостійкість та його можливо оптимізувати).

Нижче в таблиці 2.1 наведені значення параметрів для удосконаленого електронного підпису Dilithium.

Таблиця 2.1 – Значення параметрів для 256, 384, 512 біт стійкості ЕП Dilithium

Набір	(N, q)	γ_1	γ_2	(k, l)	η	β	d	h	ω
256	(256, 8380417)	523776	261888	(9,8)	2	144	14	60	184
384	(512, 8380417)	523776	261888	(7,5)	5	100	13	77	286
512	(512, 8380417)	523776	261888	(9,8)	2	74	13	118	368

Ймовірність повтору циклу при цьому складає для 256 біт – 0,15442678312246608, для 384 біт – 0,15609624568669475 і для 512 біт – 0,15247678668181552.

Виходячи з наведеного в цьому підрозділі матеріалу, нижче в таблиці 2.2 наведені значення системних параметрів ЕП для різних режимів роботи «Вершина 1».

Таблиця 2.2 – Значення системних параметрів ЕП для режимів роботи Вершина 128/64, 256/128, 384/192, 512/128 біт стійкості відповідно до класичного та квантового криптоаналізу

Набір	(N, q)	γ_1	γ_2	(k, l)	η	β	d	h	ω
128/64 (mode = 0)	(256, 8380417)	523776	261888	(5,4)	5	275	14	60	96
256/128 (mode = 1)	(256, 8380417)	523776	261888	(9,8)	2	120	14	60	184
384/192 (mode = 2)	(512, 8380417)	523776	261888	(7,5)	5	100	13	77	286
512/256 (mode = 3)	(512, 8380417)	523776	261888	(9,8)	2	74	13	118	368
256/128 (mode = 1) Dilithium	(256, 8380417)	2^{17}	$\frac{q-1}{88}$	(4,4)	2	78	13	39	80

В таблиці 2.3 наведено результати оцінки криптостійкості для удосконаленого ЕП Dilithium (в бітах) з використанням параметрів з таблиці 2.1.

Таблиця 2.3 – Оцінки криптостійкості для удосконаленого ЕП Dilithium

Набір	Primal Attack (класична)	Primal Attack (квантова)	Dual Attack (класична)	Dual Attack (квантова)	SIS (класична)	SIS (квантова)
256	298	270	296	269	293	266

Продовження таблиці 2.3 – Оцінки криптостійкості для удосконаленого ЕП Dilithium

384	440	399	438	397	503	456
512	582	527	579	525	590	535

2.2 Генерація загальносистемних параметрів ЕП Falcon для 256, 384, 512 («Вершина 2»)

В основу механізму ЕП Falcon покладено перетворення фреймворк GPV [33], в якому відкритим ключем є базис $A \in \mathbb{Z}_q^{n \times m}$ решітки Λ. У якості закритого ключа використовується редукований базис $B \in \mathbb{Z}_q^{m \times m}$ дуальної решітки $\Lambda^{\perp}$. В механізмі ЕП Falcon використовуються NTRU решітки виду [34]:

$$A = \begin{bmatrix} 1 & h \\ 0 & q \end{bmatrix}, B = \begin{bmatrix} f & g \\ F & G \end{bmatrix}, \tag{2.10}$$

де ключові дані f, g, h, F, G є поліномами у кільці поліномів $\mathbb{Z}_q[X]/(\phi(X))$. Щодо них виконується порівняння (2.11)

$$fG - gF = q \ mod \ \phi. \tag{2.11}$$

Задача вирішення рівняння (2.11) стає складною, якщо f, g є малими поліномами. Використання структурованих решіток дає змогу зменшити розмір ключів та передавати не всю матрицю базису, а лише поліноми, що її формують.

При виробленні ЕП для повідомлення m застосовується пара поліномів (s_1, s_2), для якої виконується рівність

$$s_1 + s_2 h = \mathrm{H}(r \parallel m) \tag{2.12}$$

де r – сіль ключа, H – деяка криптографічна геш-функція, що відображає бінарні данні на вектори. Таких (s_1, s_2) існує багато і знайти їх неважко, проте якщо ввести обмеження $\|s\| < \beta$, де β має достатньо мале значення, то задача стає важкою і зводиться до проблеми SIS. Таким чином, основною проблемою, на якій ґрунтується безпека механізму ЕП Falcon, є SIS на NTRU-решітках, тобто знаходження короткого цілого рішення (Short Integer Solution) [38].

З атаки на відновлення приватного ключа за допомогою відкритого ключа, що описується для ЕП Falcon у [38] для генерації загальносистемних параметрів слід знати наступне. Згідно з [35] найменший вектор може бути знайдений, якщо його проекція на простір, що натягнутий на перші B векторів $b_1^*, b_2^*, \dots, b_B^*$ буде менша за b_{2n-B}^*. Згідно з [35, 36] ця проекція може бути оцінена як

$$\sqrt{\gamma_B} * \det\left(\Lambda_{[b_1^*,\dots,b_B^*]}\right) \approx \sqrt{\frac{3}{4}} * \sigma' \sqrt{B} = \sqrt{\frac{3}{4}} * \sqrt{\frac{qe}{2}} * \frac{1}{\sqrt{2n}} * \sqrt{B} \qquad (2.13)$$

Водночас, згідно з [35, 36] b_{2n-B}^* може бути оцінений як

$$\begin{aligned} \|b_{2n-B}^*\| &\approx GH(B)^{\frac{2n+1-2(2n-B)}{2(B-1)}} \det(\Lambda)^{\frac{1}{2n}} = GH(B)^{\frac{2B-2n+1}{2B-2}} \sqrt{q} \\ &= \left(\frac{B}{2\pi e}\right)^{\frac{2B-2n+1}{2B-2}} \sqrt{q} \end{aligned} \qquad (2.14)$$

У цілому, якщо підсумувати наведене вище, то знайти параметри n, q, β можна з системи нерівностей (2.15) та умов (2.10) – (2.14):

$$\begin{cases} \left(\frac{B}{2\pi e}\right)^{\frac{2n-1}{2B-2}} \sqrt{q} < \beta \\ \left(\frac{B}{2\pi e}\right)^{\frac{2B-2n+1}{2B-2}} < \sqrt{\frac{3eB}{8n}} \end{cases} \qquad (2.15)$$

де параметр β визначається як $\beta = 1.2 * \sigma\sqrt{2nq}$, якщо використовується поліном $x^n + 1$ і $\beta^2 = \frac{(1.2*\sigma*2n\sqrt{q})^2}{n}$, якщо використовується поліном $x^n - x^{\frac{n}{2}} - 1$.

На основі (2.15) розроблено програмне забезпечення, з використанням якого були обчислені параметри n, q та β^2 ЕП Falcon для відповідних поліномів для 256, 384, 512 біт безпеки, що наведені в таблиці 2.4.

Таблиця 2.4 – Основні загальносистемні параметри ЕП Falcon («Вершина 2») для 256, 384, 512 біт безпеки

Безпека	$\phi(x)$	n	q	β^2
256	$x^n + 1$	1024	12289	87070769
384	$x^n - x^{\frac{n}{2}} - 1$	1536	18433	174141539
512	$x^n + 1$	2048	12289	200928983

Отриману криптостійкість наведено в таблиці 2.5. Криптостійкість наведено у форматі «стійкість» λ/розмір блоку.

Таблиця 2.5 – Оцінки криптостійкості для ЕП Falcon («Вершина 2»)

Безпека	Стійкість до відновлення ключа (класична)	Стійкість до відновлення ключа (квантова)	Стійкість до підробки підпису (класична)	Стійкість до підробки підпису (квантова)
256	273/936	248/936	269/922	244/922
384	413/1417	375/1417	430/1474	390/1474
512	554/1899	503/1899	599/2053	544/2053

Оцінки криптостійкості отримувалися з розміру блоку як 0,265B та 0,292B. Такий підхід вважається класичним і використовувався розробниками

Dilithium та розробниками Falcon. Проте, оцінки є досить грубими. Якщо використати підхід як в qTesla, то значення криптостійкості при тих же самих параметрах буде суттєво більшим. В таблиці 2.5 для квантових атак для 256 і 512 sn отримані значення є трохи меншими за необхідні, проте через грубість оцінки можна вважати, що вони досягають потрібного порога. Для 256 біт згенеровані параметри співпадають із параметрами, згенерованими розробниками Falcon для 256 біт рівня криптостійкості.

2.3 Висновки за розділом

1) Проєкти стандартів «Вершина 1» та «Вершина 2» розроблені на основі алгебраїчних решіток з відхиленням та алгебраїчних решіток з використанням спеціальних даних відповідно. Стандарти «Вершина 1» та «Вершина 2» базуються на математичних методах схем електронного підпису Crystals-Dilithium та Falcon відповідно.
2) У підрозділі 2.1 наводяться особливості генерації загальносистемних параметрів ЕП Dilithium («Вершина 1») для 128, 256, 384, 512 біт стійкості. Значення системних параметрів ЕП для режимів роботи Вершина 128/64, 256/128, 384/192, 512/128 біт стійкості відповідно до класичного та квантового криптоаналізу наводяться у таблиці 2.2, а також в таблиці 2.3 наводяться оцінки криптостійкості.
3) У підрозділі 2.2 наводяться особливості генерації загальносистемних параметрів ЕП Falcon для 256, 384, 512 («Вершина 2»). У таблиці 2.4 представлені основні загальносистемні параметри ЕП Falcon («Вершина 2») для 256, 384, 512 біт безпеки. У таблиці 2.5 наведені оцінки криптостійкості для ЕП Falcon («Вершина 2»), криптостійкість наведено у форматі «стійкість» λ/розмір блоку.

3 ПОРІВНЯННЯ ЕП CRYSTALS-DILITHIUM, «ВЕРШИНА 1» ТА «ВЕРШИНА 2»

У цьому розділі будуть порівняні схеми електронних підписів, які розглядалися у двох попередніх розділах. Будуть наведені результати цих порівнянь, зокрема в контексті Блокчейн.

3.1 Методики оцінки та результати порівняння постквантових алгоритмів ЕП на алгебраїчних решітках

Теоретичні дослідження в криптології направлені на оцінку, порівняльний аналіз та удосконалення існуючих та перспективних асиметричних криптоперетворень типу АСШ, ПІК, ЕП, а також симетричних криптоперетворень типу БСШ та ПСШ. Для оцінки та порівняльного аналізу вказаних асиметричних криптоперетворень застосовується комплексна методика, що включає методику оцінки та порівняння по безумовним критеріям, по умовним критеріям, а також прагматичним критеріям.

В таблиці Б.4 наведені характеристики обраних для порівняння алгоритмів (значення швидкості криптоперетворень та генерації ключів наведено в тактах). В порівнянні приймали участь проєкти стандартів «Вершина 1» та «Вершина 2», а також алгоритм Dilithium, який за попередніми дослідженнями мав кращі результати серед постквантових алгоритмів підпису, що засновані на перетвореннях на алгебраїчних решітках. Стійкість алгоритмів «Вершини» 128 біт відповідає третьому рівню стійкості NIST, 256 біт NIST – п’ятому, тому пропорційно для виконання порівняння згідно шкали оцінок попарного порівняння параметрам 384 біт був наданий шостий рівень, а 512 біт – сьомий.

В таблиці Б.5 наведені результати досліджень – відносна перевага алгоритмів ЕП, що розглядаються. Дані результати отримані методом попарних порівнянь за кожною з характеристик [39].

Нижче наведено рисунок 3.1, на якому зображено графік загальної відносної переваги алгоритмів ЕП, що розглядаються, з урахуванням вагових коефіцієнтів характеристик.

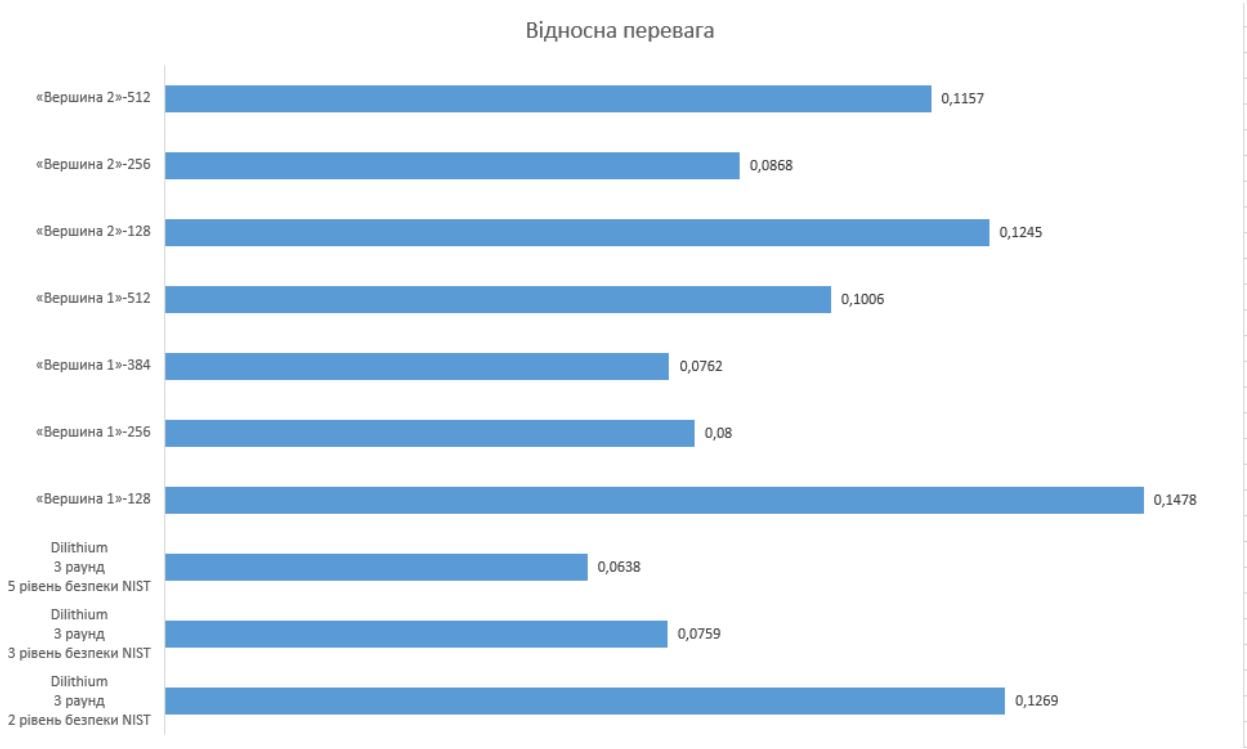

Рисунок 3.1 – Загальна відносна перевага алгоритмів ЕП (гістограма)

Як видно з рисунку 3.1 найбільшу перевагу має алгоритм «Вершина 1» з параметрами стійкості 128 біт, для більш стійких параметрів перевага вже у алгоритму «Вершина 2».

Таким чином, зроблено порівняння алгоритмів, що пройшли до третього етапу NIST, а також алгоритмів проектів стандарту «Вершина 1» та «Вершина 2». При порівнянні використовувалися два методи порівняння для отримання більш точної оцінки в залежності від вимог до алгоритмів ЕП.

3.2 Загальний огляд Блокчейн для тестування ЕП

Hyperledger Fabric — це відкрите та гнучке рішення для створення ексклюзивних платформ розподіленого реєстру. Він підтримує модульні консенсусні протоколи, що дозволяє вибирати різні моделі довіри та компроміси продуктивності. Контроль доступу та керування ідентифікацією

по суті покладається на облікові дані, видані центром сертифікації (CA) постачальника послуг членства (MSP), який, у свою чергу, знаходиться під кореневим CA, який може бути створений як Fabric-CA або зовнішній CA. Екземпляр MSP за замовчуванням покладається на інтерфейс постачальника криптографічних послуг Blockchain (BCCSP), який обробляє лише стандартні методи PKI для автентифікації, враховуючи в основному класичні підписи RSA та ECDSA. Крім того, облікові дані, видані MSP, використовують лише одну схему підпису, що робить функції, пов’язані з обліковими даними, сильно пов’язаними з окремими класичними стандартними примітивами. На жаль, добре відомо, що RSA та ECDSA вразливі до квантових атак, і поточний процес постквантової стандартизації, який проводить NIST, має на меті визначити квантово-безпечні заміни для таких криптографічних примітивів за кілька років.

Тож у наступному підрозділі будуть наведені результати використання переробки процедур керування обліковими даними та відповідних специфікацій, щоб включити деякі з цифрових підписів розглянутих вище в роботі (тобто захист як від класичних, так і від квантових атак). Реалізація Fabric інтегрується з бібліотекою Open Quantum Safe, таким чином використовується концепція криптографічної гнучкості, яка дозволяє підключати різні алгоритми до облікових даних MSP і виконувати з ними порівняльні тести.

Розробка, затвердження та впровадження постквантових криптографічних стандартів — це лише перший крок. Існує також грандіозне завдання інтеграції цих алгоритмів у широкий спектр існуючих систем, які залежать від класичної криптографії з відкритим ключем і сильно пов’язані з нею. Сама інтеграція стане новим випробуванням цих алгоритмів, особливо тому, що вони інтегровані у високопродуктивні системи. Крім того, під час інтеграції ці системи повинні залишатися захищеними від класичних атак, навіть перед обличчям потенційних недоліків або змін алгоритмів постквантової криптографії. Цей сценарій вимагає подвійного рівня захисту

від класичних і квантових атак, що може бути досягнуто за допомогою гібридних підписів, тобто одного класичного підпису та одного постквантового.

Визначним класом додатків, які повинні бути першими запроваджувачами цих алгоритмів, є розподілений реєстр Блокчейн. Технологія Блокчейн фундаментально покладається на криптографічні примітиви, особливо автентифікацію, для перевірки походження транзакцій, які мають бути додані до ланцюжка. У своїй роботі ми зосереджуємось на Hyperledger Fabric, ексклюзивному Блокчейн, який використовується у виробничих системах.

На додаток до потреби Блокчейн здійснити цей постквантовий перехід раніше, є кілька типових особливостей Блокчейну, які роблять його суворим випробувальним майданчиком для нових алгоритмів криптографії. Розгорнуті в промисловості Блокчейни мають високу пропускну здатність, а останні дослідження Hyperledger Fabric досягли 20 000 транзакцій на секунду [37]. Кожна пропозиція транзакції разом із усім етапом обробки в мережі містить підпис, а шлях коду для фіксації (комміту) кожної транзакції включає алгоритми підписання та перевірки. Таким чином, розмір підпису та швидкість алгоритму мають прямий і значний вплив на продуктивність системи. Крім того, як великі розподілені системи, які не можуть ні забезпечити синхронне оновлення, ні впоратися з тривалим простоєм для міграції, Блокчейни є одними з більш складних варіантів використання для постквантової міграції. Отримані уроки підтримки продуктивності та зворотної сумісності в цьому контексті можуть бути широко застосовані, оскільки інші системи пізніше інтегруються з постквантовими алгоритмами.

Для дослідження буде використано розробку PQFabric з [40], який є першою версією ексклюзивного корпоративного Блокчейну Hyperledger Fabric, сигнатури якого захищені як від загроз класичних, так і від квантових обчислень. Дана проектна пропозиція має зворотну сумісність для легкої міграції та достатньо гнучка для підтримки різних постквантових

криптографічних схем, що робить її готовою до майбутніх стандартів NIST та змін у реалізації OQS. Далі, за допомогою порівняльних тестів екземпляра Fabric, надається базове порівняння продуктивності між транзакціями, які використовують класичну та гібридну квантово-класичну криптографію, виявивши, що залежно від використовуваного алгоритму підпису PQFabric може досягти різної порівнянної продуктивності.

Варто підкреслити, що підпис і довжина відкритого ключа, а не час виконання алгоритму підпису, є основними факторами сповільнення PQFabric. Тож загалом дана реалізація пропонується як тестовий стенд для майбутньої роботи, зосередженої на покращенні пропускної здатності транзакцій Блокчейну за допомогою алгоритмів постквантової криптографії.

Далі наводиться набір визначень, які використовуються в контексті Fabric Блокчейн:

- Актив (Asset). Коли транзакція виконується в Блокчейні Fabric, це просто призведе до зміни стану активу. Актив є загальним визначенням і це може бути будь-що з грошовою оцінкою.
- Консорціум (Consortium). Складається з набору організацій, які хочуть вести бізнес через загальний дозволений блокчейн.
- Постачальник послуг членства (MSP). MSP — це абстракція для керування обліковими даними/ідентифікацією. Він відповідає за видачу сертифікатів одноранговим користувачам, замовникам і користувачам і, таким чином, надає їм належну автентифікацію та авторизацію. Діри та доступ можна реалізувати за допомогою політик, пов’язаних із типами сертифікатів. Він також надає функцію цифрового підпису для всіх суб’єктів.
- Fabric-CA. Це центр сертифікації, який містить кореневий сертифікат і видає сертифікати, як правило, MSP. Зокрема, Fabric-CA відповідає за видачу сертифікатів кожному MSP у консорціумі. Кожен MSP зазвичай пов'язаний з однією організацією. Сертифікат MSP

дозволяє видавати сертифікати реєстрації користувачам відповідної організації. Сертифікати реєстрації надають користувачам доступ для виклику доступних функцій смарт-контракту в Блокчейні.

- Клієнт або Користувач. Клієнт або користувач — це суб'єкт, що володіє обліковими даними (включаючи цифрові сертифікати) до певної мережі Блокчейнів і може подавати пропозиції щодо транзакцій одноранговим партнерам. Користувач також несе відповідальність за збір достатньої кількості схвалень (кількість визначається в політиці Блокчейну) від однолітків (peers) для кожної пропозиції транзакції та повторно передає їх до служби замовлення.
- Рівний або індосант, рівний або індосант (Peer or endorsing peer or endorser). Відповідальність однолітків (peers) включає перевірку підписів користувачів у пропозиціях транзакцій, виконання та перевірку пропозиції відповідно до попередньо визначених політик (узгоджених для консорціуму), схвалення та цифровий підпис кінцевого результату перевіреної пропозиції, а також повідомлення користувача про результат. Однорангові вузли також отримують блок-кандидатів від замовників і виконують остаточну перевірку та додавання до реєстру.
- Замовник (Orderer). Після отримання набору транзакцій (кількість транзакцій, які мають пройти в блоці, яку можна налаштувати), перевіряє всі підписи індосантів і запускає протокол консенсусу, щоб упорядкувати транзакції, які будуть частиною блоку-кандидату. Замовники також підписують обчислений блок-кандидат і надсилають його одноранговим клієнтам для остаточної перевірки та включення до реєстру.
- Пропозиція угоди (Transaction Proposal). Користувачі подають пропозиції транзакцій до мережі Блокчейн, щоб змінити стан своїх активів у реєстрі. Пропозиція включає в себе ідентифікатор

користувача, функцію ланцюжкового коду (смарт-контракт), яку потрібно виконати, і пов’язані з нею параметри, ідентифікатор транзакції, серед іншого.

- Транзакція. Коли користувач збирає достатню кількість схвалень для пропозиції транзакції, він збирає їх разом із пропозицією та підписує все, перш ніж надіслати його до служби замовлення. Надісланий вміст до служби замовлення називається транзакцією.

Наведені вище визначення допомагають визначити основні функції, які вимагають використання (класичних) цифрових підписів і які можуть бути оновлені гібридним аналогом.

3.3 Оцінка та результати тестування ЕП в контексті Блокчейн

Оцінка реалізації в мережі спирається на [40] та складається з одного замовника та двох однорангових вузлів, кожен з яких працює на різних машинах. Клієнт (сам на четвертій машині) надсилав усі транзакції одному вузлу, тоді як другий вузол також був доступний як індосант.

Ланцюговий код такий самий, як той, що використовується в [37], який надає прості балансові рахунки та дозволяє переказувати значення з одного рахунку на інший. Таким чином, одна транзакція стосується рівно двох рахунків. У експериментальній системі було 20 000 облікових записів, і кожен контрольний тест запустив 10 000 транзакцій, розділених на блоки розміром 100, зі 100 блоками, надісланими з кожного з 10 паралельних потоків. Транзакції були налаштовані таким чином, щоб ніхто не торкався одних і тих самих облікових записів, щоб суперечка з базою даних не була фактором. Політика схвалення вимагала лише одного партнера для схвалення транзакції. Еталонний тест вимірював час стіни на одноранговому пристрої між отриманням блоку транзакцій і фіксацією цього блоку. Для кожного раунду контрольних тестів відрізалися перші та останні кілька блоків у аналізі даних для збільшення та зменшення.

Базова криптографічна установка підписує транзакції лише за допомогою ECDSA [40]. Потім порівнювалося це з тим самим тестом із вузлами, налаштованими для підпису та перевірки за допомогою гібридних схем, що об’єднують ECDSA5 із Crystals-Dilithium-2 та «Вершина 2»-512.

Як показано на рисунку 3.2, лише ECDSA має найнижчу затримку блоку (медіана 49 мс), за нею йде гібрид ECDSA+«Вершина 2»-512 (медіана 60 мс), потім ECDSA+Dilithium2 (медіана 68 мс). Їхня середня пропускна здатність, показана на рисунку 3.3, відповідає протилежній моделі з пропускною здатністю 2084, 1788 і 1545 транзакцій на секунду відповідно. Ця загальна тенденція є очікуваною, враховуючи порівняльний аналіз цих постквантових алгоритмів.

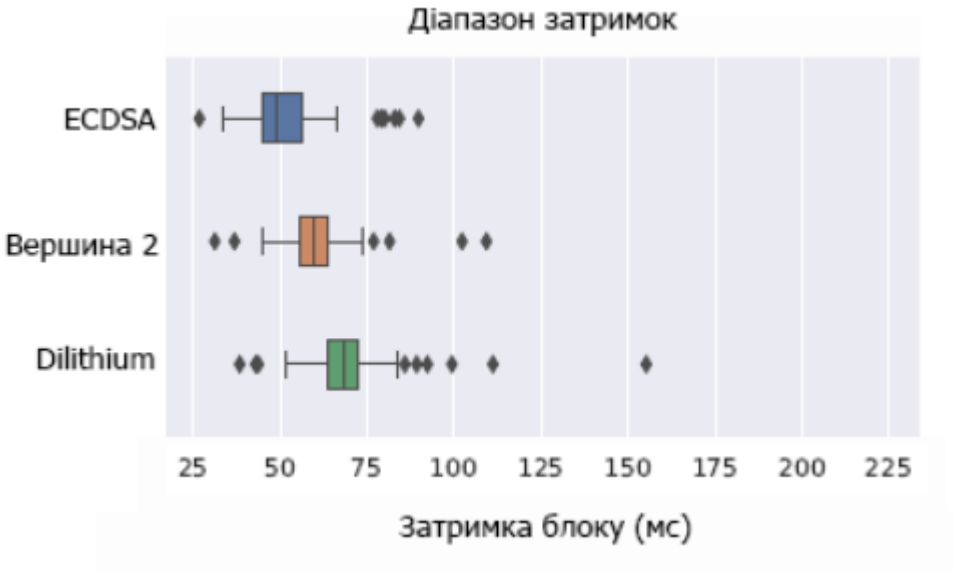

Рисунок 3.2 – Діапазон затримок для кожного блоку

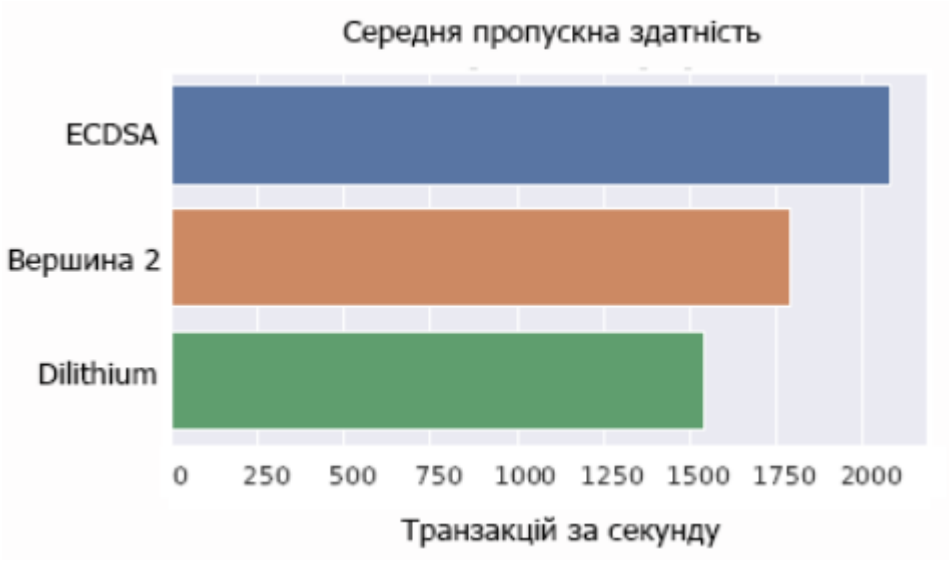

Рисунок 3.3 – Середня пропускна здатність у транзакціях за секунду

Продуктивність

Додатково слід звернути увагу на джерело уповільнення, вивчивши репрезентативні профілі ЦП однорангового вузла нижче.

Загалом було виявлено, що PQFabric, хоча й має вищу затримку блоків і нижчу пропускну здатність, ніж класична Fabric, не зазнає серйозного зниження продуктивності залежно від алгоритму постквантового підпису, який використовується. Пропускна здатність гібридної схеми «Вершина 2»-512 зменшилася в середньому лише на 14% порівняно з чистою схемою ECDSA. Дійсно, «Вершина 2»-512 сама по собі швидше, ніж ECDSA, оскільки значна частина уповільнення пов’язана з необхідністю двічі підписувати та перевіряти. Важливо та в одночас несподівано те, що очікувалося, що гібридні підписи можуть серйозно уповільнити транзакції, але гібридна схема ECDSA+«Вершина 2»-512 здається придатною для використання без додаткової оптимізації продуктивності.

Далі можна розглянути репрезентативні профілів ЦП для порівняння кожної схеми підпису. Дві функції, які спостерігають найбільше збільшення частки ЦП для гібридних схем, це $oqs.\{Sign|Verify\}$ і $sw.Hash$. Загалом очікується, що буде додатковий час виконання, необхідний для виконання

алгоритму постквантового підпису на додаток до алгоритму ECDSA, але більше дивує час, проведений у $Hash$. Частину уповільнення можна пояснити повільнішим, більшим гешем, необхідним для постквантових підписів, однак це не до кінця пояснює відмінностей між постквантовими алгоритмами. Є два основних виклики $Hash()$, які впливають на процесорний час. Перший – з перевірки підпису в MSP. Геш-функція викликається в отриманому повідомленні. Другим абонентом є $VerifyBlock$, зазвичай із служби пліток (gossip service). У цьому випадку геш-функція викликається для всього блоку. Цей блок містить як підписи, так і відкриті ключі всіх отриманих індосаментів. Для постквантових сигнатур вони можуть бути досить великими, що значно збільшує час процесора, витрачений на гешування. У таблиці 3.1 показано відсоток часу, витраченого кожною схемою підпису на гешування протягом усього контрольного тесту.

Таблиця 3.1 – Відсоток часу виконання, витраченого на Liboqs (Sign, Verify) і гешування

	Час liboqs	Час гешування
ECDSA	0%	7%
Вершина 2	22%	8%
Dilithium	6%	18%

Примітно, що швидкість алгоритму підпису насправді лише незначним чином впливає на зниження продуктивності в схемах із більшим розміром ключа та підпису. Ми бачимо, що для Hyperledger Fabric підпис і розмір відкритого ключа є прямим і значним компромісом продуктивності для швидких алгоритмів підпису, оскільки ключ і підпис самі повинні бути гешовані в блоці. Здається, найбільший вплив на продуктивність PQFabric полягає в тому, що «Вершина 2» має малий розмір ключа та підпису.

Ще однією відмінністю продуктивності між схемами підпису є діапазон затримок блоків. ECDSA, «Вершина 2» і Dilithium мали відносно стабільну продуктивність у блоках. Схеми мали стандартні відхилення 11,0, 10,0 і 13,6 відповідно.

Як згадувалося вище, гешування є значним фактором уповільнення гібридної постквантової структури, яка регулюється розміром ключа та підпису. Однак найпростішою пропозицією щодо покращення затримки блоку та пропускної здатності транзакцій для PQFabric може бути розпаралелювання постквантової та класичної перевірки. У таблиці 3.2 показується очікуваний максимальний приріст продуктивності від цього. Для цього обчислюється відсоток часу виконання, витраченого на коротшу з двох перевірок (алгоритм PQ проти ECDSA), і виводиться відсоток прискорення, очікуваного за рахунок зменшення цього часу виконання. Зауважте, що ці профілі є лише репрезентативною вибіркою, тому відсоток прискорення слід сприймати лише як приблизну інтуїцію.

Таблиця 3.2 – Очікуване прискорення від розпаралелювання постквантової і класичної перевірки

	Очікуване прискорення
«Вершина 2»	2%
Dilithium	4%

Як бачимо, очікуване прискорення є відносно скромним. Dilithium, який мав відносно високу загальну продуктивність, має отримати найбільшу користь, оскільки його час виконання був найбільш схожим на ECDSA. Таким чином, розпаралелювання було б найбільш ефективним для нього.

3.4 Висновки за розділом

1) У таблицях Б.4 та Б.5, а також на рисунку 3.1 наведені результати порівняння проєктів стандартів ЕП «Вершина 1» та «Вершина 2», а

також алгоритму Dilithium. Кращі результати серед постквантових алгоритмів ЕП, що засновані на перетвореннях на алгебраїчних решітках, мають проєкти ЕП «Вершина 1» та «Вершина 2».

2) Кращі показники у алгоритмів «Вершина 1» та «Вершина 2» («Вершина 2» на даному етапі програє через свою низьку швидкодію, але якщо її реалізація буде більш оптимізована, то вже «Вершина 2» буде на першому місці).
3) У даному розділі були також наведені основні відомості про Hyperledger Fabric, що є відкрити та гнучким рішенням для створення ексклюзивних платформ розподіленого реєстру та може використовуватися, як платформа для тестування електронних підписів у Блокчейн.
4) Як показано на рисунку 3.2, лише ECDSA має найнижчу затримку блоку (медіана 49 мс), за нею йде гібрид ECDSA+«Вершина 2»-512 (медіана 60 мс), потім ECDSA+Dilithium2 (медіана 68 мс). Їхня середня пропускна здатність, показана на рисунку 3.3, відповідає протилежній моделі з пропускною здатністю 2084, 1788 і 1545 транзакцій на секунду відповідно. Ця загальна тенденція є очікуваною, враховуючи порівняльний аналіз цих постквантових алгоритмів.
5) Загалом було виявлено, що PQFabric, хоча й має вищу затримку блоків і нижчу пропускну здатність, ніж класична Fabric, не зазнає серйозного зниження продуктивності залежно від алгоритму постквантового підпису, який використовується. Пропускна здатність гібридної схеми «Вершина 2»-512 зменшилася в середньому лише на 14% порівняно з чистою схемою ECDSA. Дійсно, «Вершина 2»-512 сама по собі швидше, ніж ECDSA, оскільки значна частина уповільнення пов’язана з необхідністю двічі підписувати та перевіряти. Важливо та в одночас несподівано те, що очікувалося, що гібридні підписи можуть серйозно уповільнити

транзакції, але гібридна схема ECDSA+«Вершина 2»-512 здається придатною для використання без додаткової оптимізації продуктивності.

6) Ще однією відмінністю продуктивності між схемами підпису є діапазон затримок блоків. ECDSA, «Вершина 2» і Dilithium мали відносно стабільну продуктивність у блоках. Схеми мали стандартні відхилення 11,0, 10,0 і 13,6 відповідно.
7) Найпростішою пропозицією щодо покращення затримки блоку та пропускної здатності транзакцій для PQFabric може бути розпаралелювання постквантової та класичної перевірки. У таблиці 3.2 показується очікуваний максимальний приріст продуктивності від цього.
8) Очікуване прискорення є відносно скромним. Dilithium, який мав відносно високу загальну продуктивність, має отримати найбільшу користь, оскільки його час виконання був найбільш схожим на ECDSA. Таким чином, розпаралелювання було б найбільш ефективним для нього.
9) «Вершина 2» згідно з результатами є найбільш актуальною ЕП для використання в такому контексті в Блокчейн. Хоча підрозділу 3.1 «Вершина 1» показує результати кращі за Dilithium, але скоріш за все «Вершина 2» буде все ще більш актуальною в Блокчейн. Однак, зважаючи на те, що при розпаралелюванні найбільший прогнозований приріст має Dilithium, то в майбутньому є сенс виконати порівняння «Вершина 1» та «Вершина 2» з використанням розпаралелювання.

ВИСНОВКИ

1) У першому розділі роботи була розглянута та описана схема Dilithium. Dilithium – схема цифрового підпису, безпека якої базується на складності пошуку коротких векторів у решітках. Детально описувалася її математична модель, принципи підписання, генерації ключів та інші алгоритми та методи на яких заснована ця схема.
2) Безпеку схеми підпису можна довести в моделі випадкового оракула (ROM) на основі жорсткості двох проблем. Перша — це стандартна проблема LWE (над поліноміальними кільцями), яка вимагає відрізнити $(A, t := As_1 + s_2)$ від (A, u), де u є рівномірно випадковим. Інша проблема полягає в тому, що в [14] було названо проблемою SelfTargetMSIS, яка є проблемою пошуку вектора $\begin{bmatrix} z \\ c \\ v \end{bmatrix}$ з малими коефіцієнтами та повідомленням (дайджестом) μ, що задовольняє
$$\mathrm{H}\left(\mu \parallel [A \mid t \mid \mathrm{I}] \cdot \begin{bmatrix} z \\ c \\ v \end{bmatrix}\right) = c.$$
3) Незважаючи на те, що проблеми MLWE та MSIS визначаються над поліноміальними кільцями, наразі не виявлено жодного способу використання цієї кільцевої структури, і тому найкращі атаки встановлюються, просто розглядаючи ці проблеми як проблеми LWE та SIS. Проблеми LWE та SIS точно такі ж, як у визначеннях MLWE та MSIS вище в роботі, де йдеться про заміну кільця R_q на $\mathbb{Z}_q$.
4) Попередній аналіз дозволяє зробити висновок, що сучасні варіанти механізмів LWE ґрунтуються на поліноміальних кільцях, зокрема на $R_q = \mathbf{Z}_q[X]/(x^{2^n}+1)$. Властивості поліномів кільця дозволяють довести ряд теоретичних тверджень щодо стійкості криптосистеми і розробляти ефективні програмні реалізації. Проте, такі кільця мають нетривіальні

підполя, що теоретично може використовуватися для криптоаналізу, проте на практиці атак, що застосовують ці додаткові структури, не було знайдено, або ці атаки знаходяться в незавершеному вигляді досліджень.

5) Проблеми криптоаналізу RLWE та MLWE по суті зводяться до LWE проблеми. Таке зведення можливо для $R_q = \mathbf{Z}_q[X]/(x^{2^n}+1)$, оскільки доведено, що RLWE є не менш стійким, ніж LWE. Проте, при такому підході внутрішня структура кільця ігнорується.
6) Безпека Dilithium базується на жорсткості проблем типу LWE та SIS. Схеми шифрування на основі решітки базуються виключно на проблемі LWE та її похідних із доданою структурою (наприклад, MLWE, NTRU тощо), природно, що проблемам типу LWE у майбутньому приділятиметься більше уваги. Хоча обидві проблеми досить схожі в тому, що вони вимагають пошуку коротких векторів у певній решітці, і, крім того, найкращі алгоритми мають однаковий основний компонент, тож розробники схеми вважають, що має сенс бути трохи більш консервативним щодо параметрів на стороні SIS.
7) Атаки на решітках полягають у зведенні проблеми LWE до достатньо вивчених теоретичних проблем в теорії решіток. Існують 3 основні підходи: зведення LWE до BDD, зведення LWE до SIS, зведення LWE до uSVP. Кожен з цих підходів в кінцевому випадку зводиться до задачі пошуку достатньо малого вектора на решітці, для чого використовується алгоритм BKZ та його варіації.
8) Точні оцінки для BKZ та його варіацій не відомі. При практичній оцінці використовується ряд еврістичних підходів та екстраполяція результатів, що отримані на решітках меншої розмірності. Це становить основну проблему при оцінці криптостійкості систем подібних Dilithium, оскільки немає гарантії, що не з’явиться кращого способу редукції решіток, або оцінка виявиться недопустимо не точною.

9) Деталізовані дві атаки називаються Primal Attack і Dual Attack відповідно і складають основу для оцінки складності криптосистем на основі LWE (та його різновидів). Всі інші покращення як правило є евристиками.
10) Атака підробки ЕП здійснюється шляхом вирішення задачі SIS, причому дуальна атака фактично є вирішенням SIS. Проте автори ЕП Dilithium запропонували інший підхід до вирішення цієї задачі. В SIS потрібно знайти вектор, всі елементи якого менші за певну величину. Автори Dilithium пропонують шукати малі вектори на решітці до тих пір, доки всі елементи такого малого вектора не будуть менші за задану величину в задачі SIS.
11) Проєкти стандартів «Вершина 1» та «Вершина 2» розроблені на основі алгебраїчних решіток з відхиленням та алгебраїчних решіток з використанням спеціальних даних відповідно. Стандарти «Вершина 1» та «Вершина 2» базуються на математичних методах схем електронного підпису Crystals-Dilithium та Falcon відповідно.
12) У другому розділі наводяться особливості генерації загальносистемних параметрів ЕП Dilithium («Вершина 1») для 128, 256, 384, 512 біт стійкості. Значення системних параметрів ЕП для режимів роботи Вершина 128/64, 256/128, 384/192, 512/128 біт стійкості відповідно до класичного та квантового криптоаналізу наводяться у таблиці 2.2, а також в таблиці 2.3 наводяться оцінки криптостійкості. Також наводяться особливості генерації загальносистемних параметрів ЕП Falcon для 256, 384, 512 («Вершина 2»). У таблиці 2.5 представлені основні загальносистемні параметри ЕП Falcon («Вершина 2») для 256, 384, 512 біт безпеки. У таблиці 2.5 наведені оцінки криптостійкості для ЕП Falcon («Вершина 2»), криптостійкість наведено у форматі «стійкість» λ/розмір блоку.
13) У таблицях Б.4 та Б.5, а також на рисунку 3.1 наведені результати порівняння проєктів стандартів ЕП «Вершина 1» та «Вершина 2», а

також алгоритму Dilithium. Кращі результати серед постквантових алгоритмів ЕП, що засновані на перетвореннях на алгебраїчних решітках, мають проєкти ЕП «Вершина 1» та «Вершина 2».

14) Кращі показники у алгоритмів «Вершина 1» та «Вершина 2» («Вершина 2» на даному етапі програє через свою низьку швидкодію, але якщо її реалізація буде більш оптимізована, то вже «Вершина 2» буде на першому місці).

15) Як показано на рисунку 3.2, лише ECDSA має найнижчу затримку блоку (медіана 49 мс), за нею йде гібрид ECDSA+«Вершина 2»-512 (медіана 60 мс), потім ECDSA+Dilithium2 (медіана 68 мс). Їхня середня пропускна здатність, показана на рисунку 3.3, відповідає протилежній моделі з пропускною здатністю 2084, 1788 і 1545 транзакцій на секунду відповідно. Ця загальна тенденція є очікуваною, враховуючи порівняльний аналіз цих постквантових алгоритмів.

16) Загалом було виявлено, що PQFabric, хоча й має вищу затримку блоків і нижчу пропускну здатність, ніж класична Fabric, не зазнає серйозного зниження продуктивності залежно від алгоритму постквантового підпису, який використовується. Пропускна здатність гібридної схеми «Вершина 2»-512 зменшилася в середньому лише на 14% порівняно з чистою схемою ECDSA. Дійсно, «Вершина 2»-512 сама по собі швидше, ніж ECDSA, оскільки значна частина уповільнення пов’язана з необхідністю двічі підписувати та перевіряти. Важливо та в одночас несподівано те, що очікувалося, що гібридні підписи можуть серйозно уповільнити транзакції, але гібридна схема ECDSA+«Вершина 2»-512 здається придатною для використання без додаткової оптимізації продуктивності.

17) Ще однією відмінністю продуктивності між схемами підпису є діапазон затримок блоків. ECDSA, «Вершина 2» і Dilithium мали відносно стабільну продуктивність у блоках. Схеми мали стандартні відхилення 11,0, 10,0 і 13,6 відповідно.

18) Найпростішою пропозицією щодо покращення затримки блоку та пропускної здатності транзакцій для PQFabric може бути розпаралелювання постквантової та класичної перевірки. У таблиці 3.2 показується очікуваний максимальний приріст продуктивності від цього.

19) Очікуване прискорення є відносно скромним. Dilithium, який мав відносно високу загальну продуктивність, має отримати найбільшу користь, оскільки його час виконання був найбільш схожим на ECDSA. Таким чином, розпаралелювання було б найбільш ефективним для нього. «Вершина 2» згідно з результатами є найбільш актуальною ЕП для використання в такому контексті в Блокчейн. Хоча підрозділу 3.1 «Вершина 1» показує результати кращі за Dilithium, але скоріш за все «Вершина 2» буде все ще більш актуальною в Блокчейн. Однак, зважаючи на те, що при розпаралелюванні найбільший прогнозований приріст має Dilithium, то в майбутньому є сенс виконати порівняння «Вершина 1» та «Вершина 2» з використанням розпаралелювання.

20) У роботі пропонується розуміння несумісності та проблем інтеграції, з якими виробничі системи можуть зіткнутися під час власного квантово-безпечного переходу. Надаються контрольні показники, які показують, що залежно від вибраного постквантового криптографічного алгоритму ЕП PQFabric може досягти затримки та продуктивності, порівнянної з класичною структурою.

21) Завдяки аналізу та порівнянню ЕП Dilithium, «Вершина 1» та «Вершина 2», зокрема в Блокчейн, вважаємо, що робота суттєво сприяє дискусії про постквантові криптографічні стандарти, і сподіваємося, що вона буде корисною як для тих, хто розробляє стандарти, так і для тих, хто прагне їх реалізувати.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "NIST Released NISTIR 8105, Report on Post-Quantum Cryptography". 21 December 2016. Retrieved 5 November 2019.
2. "Post-Quantum Cryptography Standardization – Post-Quantum Cryptography". Csrc.nist.gov. 3 January 2017. Retrieved 31 January 2019.
3. Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process / Alagic G. and others. NIST IR 8413-upd1. 29 September 2022.
4. Lattice signatures and bimodal gaussians / L. Ducas, A. Durmus and others – CRYPTO, 2013. – P. 40-56.
5. Efficient identity-based encryption over NTRU lattices / Ducas L., Lyubashevsky V. and others – ASIACRYPT, 2014. – P. 22-41.
6. Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme / Bruinderink L. G., Hülsing A. and others – CHES, 2016. P. 323–345.
7. Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongswan and electromagnetic emanations in microcontrollers / Thomas Espitau, Pierre-Alain Fouque and others – CCS, 2017. P. 1857-1874.
8. To BLISS-B or not to be: Attacking strongswan’s implementation of post-quantum signatures / P. Pessl, L. G. Bruinderink, and others – CCS, 2017. P. 1843-1855.
9. Lyubashevsky V. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures / Lyubashevsky V. – ASIACRYPT. 2009. P. 598-616.
10. Practical lattice-based cryptography: A signature scheme for embedded systems / Güneysu T. and others. – CHES, 2012. P. 530-547.
11. Lyubashevsky V. Lattice signatures without trapdoors / Lyubashevsky V. // EUROCRYPT – 2012 – № 33 – P. 738-755.

12. Bai S. An improved compression technique for signatures based on learning with errors / Bai S., Galbraith S. D. – CT-RSA, 2014. P. 28–47.
13. The lattice-based digital signature scheme qtesla / Alkim E., Barreto P. S. L. M., Bindel N. and others. Cryptology ePrint Archive, Report 2019/085, 2019. URL: https://eprint.iacr.org/2019/085 (дата звернення: 25.10.2022).
14. Kiltz E. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. / Kiltz E., Lyubashevsky V., Schaffner C. // *EUROCRYPT*. International Conference on the Theory and Application of Cryptographic Techniques. Saragossa : – Spain. 2018. – P. 552-586.
15. Breaking ed25519 in wolfssl / Samwel N. and others. – CT-RSA, 2018. P. 1-20.
16. Attacking deterministic signature schemes using fault attacks / Poddebniak D. and others. – EuroS&P, 2018. P. 338-352.
17. Pointcheval D. Security arguments for digital signatures and blind signatures / Pointcheval D. and Stern J – J. Cryptology, 2000. P. 361-396.
18. Bellare M. Multi-signatures in the plain public-key model and a general forking lemma / Bellare M., Neven G. // *ACM*. Conference on Computer and Communications Security 2006, Alexandria Virginia: USA. 2006. P. 390-399.
19. Security of the fiat-shamir transformation in the quantum random-oracle model / Don J. and others. Cryptology ePrint Archive, Report 2019/190, 2019. URL: https://eprint.iacr.org/2019/190 (дата звернення: 03.11.2022).
20. Crystals-dilithium: A latticebased digital signature scheme / Ducas L. and others – IACR Trans. Cryptogr. Hardw. Embed. Syst., 2018. P. 238-268.
21. Revisiting post-quantum fiat-shamir / Liu Q. and Zhandry M. Cryptology ePrint Archive, Report 2019/262, 2019. URL: https://eprint.iacr.org/2019/262 (дата звернення: 03.11.2022).
22. Knuth D. The Art of Computer Programming, volume 2. Addison-Wesley, 3 edition, 1997. 145 p.

23. Postquantum key exchange – a new hope / Alkim E. and others – *Proceedings of the 25th USENIX Security Symposium*, USENIX Association, 2016. 327-343 p. URL: http://cryptojedi.org/papers/#newhope (дата звернення: 05.11.2022).
24. Schnorr C. P. / Efficient identification and signatures for smart cards – CRYPTO, 1989. P. 239–252.
25. Quantum attacks on classical proof systems: The hardness of quantum rewinding / Ambainis A. and others – FOCS, 2014. P. 474-483.
26. CRYSTALS-Dilithium. Algorithm Specifications and Supporting Documentation / Bai S. and others, 1 October 2020. P. 1-38.
27. New directions in nearest neighbor searching with applications to lattice sieving. / Becker A. and others – *SODA'16 Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete Algorithms*, SIAM, 2016. P. 10-24. URL: https://eprint.iacr.org/2015/1128 (дата звернення: 04.11.2022).
28. Laarhoven T. / Search problems in cryptography – PhD thesis, Eindhoven University of Technology, 2015. 27 p.
29. CRYSTALS–Dilithium / Lyubachevsky V. and others – Techn. rep. NIST, 2017. URL: https://crc.nist.gov./projects/post-quantum-cryptogtraphy/ round-1-submissions (дата звернення: 06.11.2022).
30. Revisiting the expected cost of solwing uSVP and applications to LWE / Albrecht M.R. and others – Cryptology ePrint Archive, Report 2017/815. URL: http://eprint.iacr.org/2017/815 (дата звернення: 06.11.2022).
31. Горбенко І. Д. Методи обчислення системних параметрів для електронного підпису «Crystals-Dilithium» 128, 256, 384 та 512 біт рівнів безпеки / Горбенко І. Д. та інші // Радіотехніка – Харківский національний університет радіоелектроніки, 2020. – Випуск 202 – С. 5-27.
32. Peikert C. How (not) to instanaite Ring-LWE / Cryptology ePrintAchive 2016/351, 2016.
33. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and newcryptographic constructions / Richard E. L. and Dwork C., – 40th ACM STOC, ACM Press, 2008. P. 197-206.

34. Falcon Round 3 Presentation. URL: https://csrc.nist.gov/presentations/2021/falcon-round-3-presentation (дата звернення: 06.11.2022).
35. Micciancio D., Walter M. Practical, predictable lattice basis reduction // Fischlin M. and Coron J. S., editors. EUROCRYPT 2016, Part I, volume 9665 of LNCS. – Vienna :Austria, May 8–12, 2016. P. 820-849.
36. Prest T. Gaussian Sampling in Lattice-Based Cryptography / Theses, École Normale Supérieure, 2015.
37. “FastFabric: Scaling hyperledger fabric to 20 000 transactions per second,” / Gorenflo C. and others, *International Journal of Network Management*, vol. 30, № 5, 2020. 2099 p. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.2099 (дата звернення: 11.11.2022).
38. Горбенко І. Д. Генерація загальносистемних параметрів для криптосистеми Falcon для 256, 384, 512 біт безпеки/ Горбенко І. Д. та інші // Радіотехніка – Харківский національний університет радіоелектроніки, 2020. – Випуск 202 – С. 57-63.
39. Основні положення та результати порівняння властивостей електронних підписів постквантового періоду на алгебраїчних решітках / Горбенко І. Д. та інші // Радіотехніка : Всеукр. міжвід. наук.-техн. зб. – Харків, 2021. – Випуск 205. – С. 5–21.
40. PQFabric: A Permissioned Blockchain Secure from Both Classical and Quantum Attacks / Das B. and others, 2021. P. 1-9.

ДОДАТОК А

Біт-пакування

Далі буде описано, як кодуються вектори як рядки байтів. Це необхідно для поглинання їх у SHAKE і визначення розташування даних ключів і підпису. Щоб зменшити час обчислень, витрачений на SHAKE, і розміри ключів і підписів, використовується бітове упаковування. Загальне правило, якого слід дотримуватися, полягає в тому, що якщо діапазон, у якому знаходиться елемент x, складається виключно з цілих невід'ємних чисел, тоді просто запаковується ціле число x. Якщо x з діапазону $[-a, b]$, який може містити кілька від'ємних чисел, тоді запаковується додатне ціле $b - x$.

Почнемо з вектора w_1. Він складається з k поліномів $\omega_1, 1, \ldots, \omega_{1,k}$ у R_q з коефіцієнтами, які є округленнями елементів у $\mathbb{Z}_q$ відносно $\alpha = 2\gamma_2$. Звідси випливає, що його коефіцієнти лежать у $\{0, \ldots, 15\}$ для рівнів 3 і 5 NIST і в $\{0, \ldots, 43\}$ для рівня NIST 2. Вони можуть бути представлені 4 і, відповідно, 6 бітами кожен. Це дозволяє упаковувати w_1 у рядок із $k \cdot 256 \cdot 4/8 = k \cdot 128$, відповідно $k \cdot 192$ байтів. Для рівнів безпеки NIST 3 і 5 кожен байт кодує два послідовних коефіцієнта полінома $w_{1,i}$ в його молодших 4 бітах і старших 4 бітах відповідно. Нижче наведено рисунок А.1 для пояснення точної упаковки бітів. Для рівня безпеки NIST 2 упаковка виглядає так, як показано на рисунку А.2.

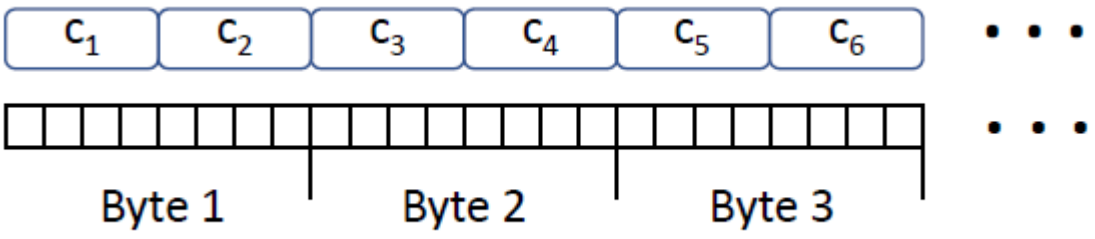

Рисунок А.1 – Упаковка бітів w_1 для рівнів 3 і 5 NIST

На рисунку А.1 видно k поліномів, що містять w_1, є $\omega_{1,1},\ldots,\omega_{1,k}$, а $c_1,\ldots,c_{256}$ — коефіцієнти $\omega_{1,1}$ (з меншими ступенями спочатку), $c_{257},\ldots,c_{512}$ — коефіцієнти $\omega_{1,2}$ тощо.

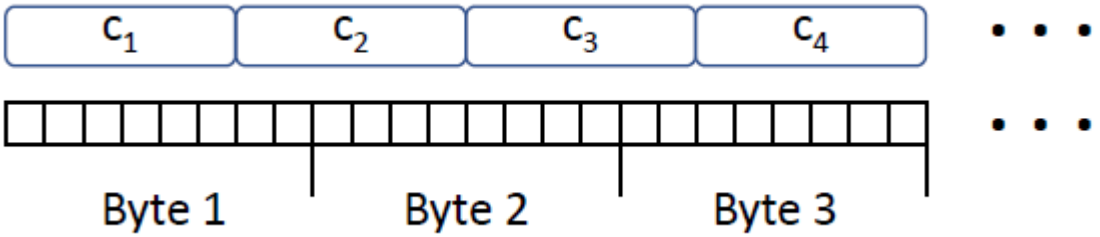

Рисунок А.2 – Упаковка бітів w_1 для рівня 2 NIST

Далі звернемося до вектора t_1, який є двоїчним округленням t. Зазначимо, що $q-1=2^{23}-2^{13}=(2^{10}-1)2^{13}$, що показує, що коефіцієнти k поліномів з t_1 лежать у $\{0,\ldots,2^{10}-1\}$ і можуть бути представлені 10 бітами кожен. Ці 10 біт на коефіцієнт, у порядку байтів у порядку байтів, розрядно упаковані. Загалом t_1 потребує $k\cdot 256\cdot 10/8=320k$ байтів. На рисунку А.3 пояснена точна упаковка бітів.

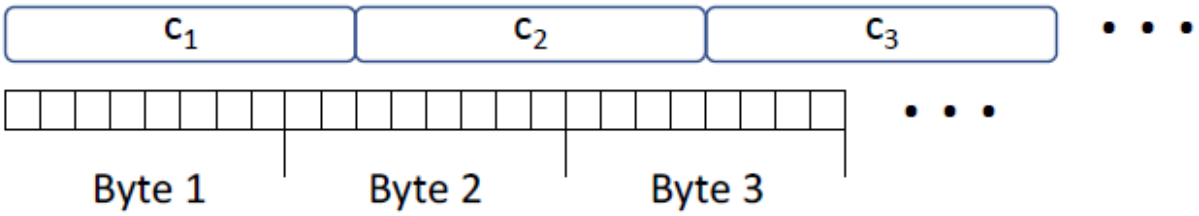

Рисунок А.3 – Упаковка бітів t_1

На рисунку А.3 k поліномів, що містять t_1, є $t_{1,1},\ldots,t_{1,k}$ і покладемо $c_1,\ldots,c_{256}$ — коефіцієнти $t_{1,1}$ (з меншими степенями спочатку), $c_{257},\ldots,c_{512}$ – коефіцієнти $t_{1,2}$ тощо.

Коефіцієнти поліномів t_0 можна записати у вигляді $2^{12} - v$ з $v \in \{0, \ldots, 2^{13} - 1\}$. Ці v у порядку байтів у порядку байтів упаковані бітами. Це призводить до $256 \cdot 13/8$ байтів на поліном і $k \cdot 256 \cdot 13/8 = 416k$ байт для t_0. На рисунку А.4 пояснена точна упаковка бітів.

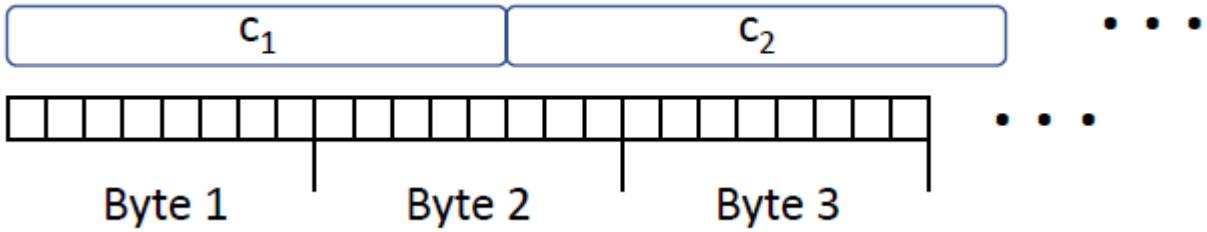

Рисунок А.4 – Упаковка бітів t_0

Поліноми від s_1 і s_2 мають коефіцієнти з нескінченною нормою не більше η. Отже, кожен коефіцієнт цих поліномів еквівалентний за модулем q до $\eta - c$ з деяким $c \in \{0, \ldots, 2\eta\}$. У пакуванні бітів значення для c зберігаються таким чином, що кожен поліном потребує $256\lceil log(2\eta + 1)\rceil/8$ байтів. Це становить $256 \cdot 3/8 = 96$ байт для рівнів 2 і 5 NIST і $256 \cdot 4/8 = 128$ байт для рівня 3. Упаковка бітів виконується подібно до випадку w_1, t_1 і t_0. На рисунку А.5 пояснена точна упаковка бітів.

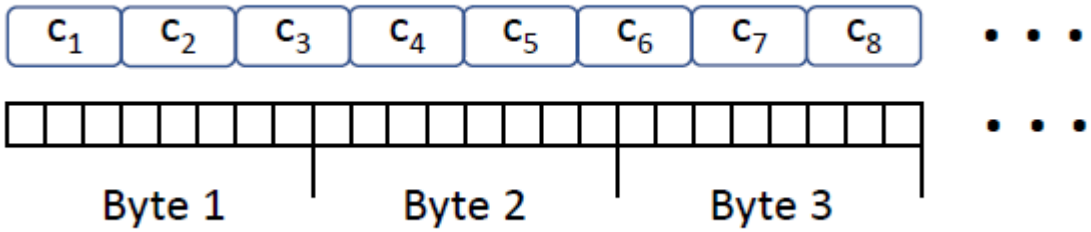

Рисунок А.5 – Упаковка бітів s_i

Нарешті, z містить поліноми, коефіцієнти яких еквівалентні за модулем q до $\gamma_1 - c$ з $c \in \{0, \ldots, 2\gamma_1 - 1\}$ і ці значення c упаковані в біти. Для рівнів 3 і

5 NIST для упаковки бітів z потрібно $l \cdot 256 \cdot 20/8 = 640l$ байтів, а блоки з 2 коефіцієнтів зберігаються в 5 послідовних байтах. На рисунку А.6 пояснена точна упаковка бітів. Для рівня NIST 2 для бітового упакування z потрібно $l \cdot 256 \cdot 18/8 = 576l$ байтів

На рисунку А.4 k поліномів, що містять t_0, є $t_{0,1}, \ldots, t_{0,k}$ де c_i визначено так, що $2^{13} - c_1, \ldots, 2^{13} - c_{256}$ – коефіцієнти при $t_{0,1}$ (спочатку з меншими ступенями), $2^{13} - c_{257}, \ldots, 2^{13} - c_{512}$ – коефіцієнти $t_{0,2}$ тощо.

На рисунку А.5 поліноми l, що містять s_1, є $s_{1,1}, \ldots, s_{1,l}$ і нехай $\eta - c_1, \ldots, \eta - c_{256}$ — коефіцієнти $s_{1,1}$ (з меншими степенями спочатку), $\eta - c_{257}, \ldots, \eta - c_{512}$ — коефіцієнти $s_{1,2}$ тощо, де $c_i \in \{0, \ldots, 2\eta\}$. k поліномів, що містять s_2, є $s_{2,1}, \ldots, s_{2,l}$ і покладемо – $\eta - c_1, \ldots, \eta - c_{256}$ – коефіцієнти $s_{2,1}$ (з меншими степенями спочатку), $\eta - c_{257}, \ldots, \eta - c_{512}$ – коефіцієнти $s_{2,2}$ і т. д. $c_i \in \{0, \ldots, 2\eta\}$. Наведене вище зображення стосується наборів параметрів, де для c_i потрібно три біти на коефіцієнт. Упаковка чотирьох бітів на коефіцієнт виконується очевидним чином і виглядає як упаковка w_i на рисунку А.1.

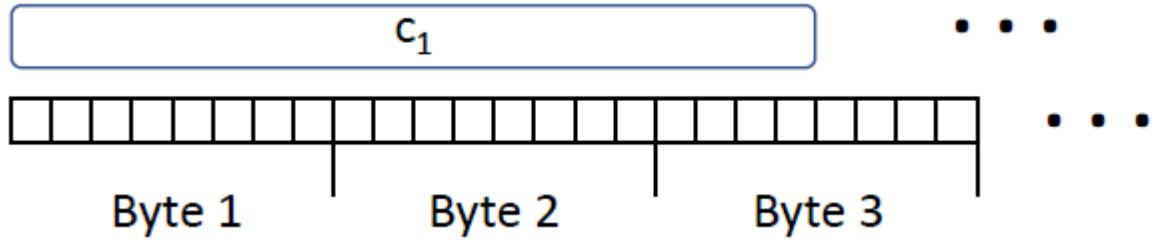

Рисунок А.6 – Упаковка бітів z

На рисунку А.6 поліноми l, що містять z, є $z_1, \ldots, z_l$ і покладемо $\gamma_1 - c_1, \ldots, \gamma_1 - c_{256}$ – коефіцієнти при z_1 (спочатку з меншими степенями), $\gamma_1 - c_{257}, \ldots, \gamma_1 - c_{512}$ – коефіцієнти z_2 тощо. На наведеному вище рисунку А.6, кожен коефіцієнт z використовує 20 бітів. Аналогічно виглядає упаковка для рівня 2, де кожен коефіцієнт становить 18 біт.

ДОДАТОК Б

Таблиця Б.1 – Виклик та розширені параметри безпеки для Dilithium

Виклик і розширені набори	1--	1-	5+	5++
q	8380417	8380417	8380417	8380417
d	10	13	13	13
вага c	24	30	60	60
виклик ентропії	135	160	257	257
γ_1	2^{17}	2^{17}	2^{19}	2^{19}
γ_2	$(q-1)/128$	$(q-1)/128$	$(q-1)/32$	$(q-1)/32$
(k, l)	(2, 2)	(3, 3)	(9, 8)	(10, 9)
η	6	3	2	2
β	144	90	120	120
w	10	80	85	90
розмір pk (байти)	864	992	2912	3232
розмір підпису (байти)	1196	1843	5246	5892
Повторення	5.2	4.87	4.59	5.48
Розмір блоку BKZ b для зламу SIS Core-SVP Classical	190 (165) 55 (49)	305 (305) 89 (89)	1055 (1005) 308 (293)	1200 (1145) 360 (334)
Розмір блоку BKZ b для зламу LWE Core-SVP Classical	200 58	305 89	1020 298	1175 343

Таблиця Б.2 – Вихідні розміри, безпека та продуктивність Dilithium

Рівень безпеки NIST	2	3	5

Продовження таблиці Б.2 – Вихідні розміри, безпека та продуктивність Dilithium

Вихідний розмір			
Розмір відкритого ключа (байти)	1312	1952	2592
Розмір підпису (байти)	2420	3293	4595
Твердість LWE (Core-SVP і покращена)			
BKZ блок-розмір b (GSA)	423	624	863
Класичний Core-SVP	123	182	252
Квантовий Core-SVP	112	165	229
BKZ блок розміром b (симуляція)	433	638	883
log_2 Класичні ворота	159	217	285
log_2 Класична пам'ять	98	139	187
Твердість SIS (Core-SVP)			
BKZ блок розміром b	423 (417)	638 (602)	909 (868)
Класичний Core-SVP	123 (121)	186 (176)	265 (253)
Квантовий Core-SVP	112 (110)	169 (159)	241 (230)
Продуктивність (неоптимізований еталонний код, Skylake)			
Gen медіанних циклів	300, 751	544, 232	819, 475
Sign медіанних циклів	1, 081, 174	1, 713, 783	2, 383, 399
Sign середніх циклів	1, 355, 434	2, 348, 703	2, 856, 803
Verify медіанних циклів	327, 362	522, 267	871, 609
Продуктивність (AVX2, Skylake)			
Gen медіанних циклів	124, 031	256, 403	298, 050
Sign медіанних циклів	259, 172	428, 587	538, 986
Sign середніх циклів	333, 013	529, 106	642, 192
Verify медіанних циклів	118, 412	179, 424	279, 936
Продуктивність (AVX2+AES, Skylake)			
Gen медіанних циклів	70, 548	153, 856	153, 936

Продовження таблиці Б.2 – Вихідні розміри, безпека та продуктивність Dilithium

Sign медіанних циклів	194, 892	296, 201	344, 578
Sign середніх циклів	251, 144	366, 470	418, 157
Verify медіанних циклів	72, 633	102, 396	151, 066

Таблиця Б.3 – Параметри Dilithium

Параметри	Значення
(N, q)	Ступінь та модуль коефіцієнтів поліномів
(k, l)	Кількість поліномів в матриці
η	Значення коефіцієнтів приватних ключів s, що знаходяться в межах $[-\eta, \eta]$
(γ_1, γ_2)	Обмеження на максимальні значення коефіцієнтів під час вироблення підпису. Сильно впливають на стійкість до атак з підробкою підпису.
β	Також впливає на максимальне значення коефіцієнтів, проте впливає не стільки на стійкість, скільки на розмір підпису. Фактично дозволяє знаходити баланс між техніко-економічними показниками та стійкістю.
d	Впливає на стійкість до атак до підробки підпису
h	Визначає об’єм простору з якого обирається challenge
w	Знаходиться з практичних міркувань. Впливає тільки на розмір підпису.

Таблиця Б.4 – Характеристики алгоритмів ЕП, що засновані на перетвореннях на алгебраїчних решітках та розглядаються в даній роботі

Алгоритми	$I_{ст.}$	$l_{в.к.}$	$l_{о.к.}$	$l_{рез.}$	$T_{пр.}$	$T_{зв.}$	$T_{гк.}$

Продовження таблиці Б.4 – Характеристики алгоритмів ЕП, що засновані на перетвореннях на алгебраїчних решітках та розглядаються в даній роботі

Dilithium 3 раунд 2 рівень безпеки NIST	2	1 312	3 504	2 420	259 172	118 412	124 031
Dilithium 3 раунд 3 рівень безпеки NIST	3	1 952	3 856	3 293	428 587	179 424	256 403
Dilithium 3 раунд 5 рівень безпеки NIST	5	2 592	5 792	4595	538 986	279 936	298 050
«Вершина 1»-128	3	1 472	3 488	2 693	133 340	109 818	90 328
«Вершина 1»-256	5	2 624	5 792	5 345	259 103	233 712	229 669
«Вершина 1»-384	7	4 528	9 088	6762	411 040	398 029	317 324
«Вершина 1»-512	9	5 824	11 008	10708	643 744	620 989	485 471
«Вершина 2»-128	3	897	4097	666	655 672	139 620	33 696 000
«Вершина 2»-256	5	1 793	8193	1 280	1 338 825	285 714	107 055 000

Продовження таблиці Б.4 – Характеристики алгоритмів ЕП, що засновані на перетвореннях на алгебраїчних решітках та розглядаються в даній роботі

«Вершина 2»-512	9	3 585	5121	2 515	2 600 053	265 416	28 493 603 229

Таблиця Б.5 – Відносна перевага алгоритмів ЕП за кожною з характеристик

Алгоритми	$I_{ст.}$	$l_{в.к.}$	$l_{о.к.}$	$l_{рез.}$	$T_{пр.}$	$T_{зв.}$	$T_{гк.}$
Dilithium 3 раунд 2 рівень безпеки NIST	0,0198	0,1770	0,1816	0,1131	0,1583	0,1857	0,2090
Dilithium 3 раунд 3 рівень безпеки NIST	0,0299	0,0965	0,1475	0,0606	0,0849	0,0984	0,1082
Dilithium 3 раунд 5 рівень безпеки NIST	0,0697	0,0655	0,0768	0,0507	0,0666	0,0506	0,0915
«Вершина 1»-128	0,0299	0,1395	0,1816	0,0800	0,3006	0,2195	0,2696
«Вершина 1»-256	0,0697	0,0655	0,0768	0,0339	0,1583	0,0716	0,1388
«Вершина 1»-384	0,1453	0,0327	0,0407	0,0261	0,0975	0,0348	0,0797
«Вершина 1»-512	0,2681	0,0233	0,0296	0,0173	0,0479	0,0218	0,0608
«Вершина 2»-128	0,0299	0,2487	0,1212	0,3211	0,0479	0,1466	0,0192
«Вершина 2»-256	0,0697	0,1108	0,0467	0,1989	0,0238	0,1130	0,0146
«Вершина 2»-512	0,2681	0,0406	0,0973	0,0984	0,0143	0,0581	0,0086

ДОДАТОК В

ВІДОМОСТІ

про автора та наукового керівника наукової роботи

«67201GOGVM»

Автор	Науковий керівник
1. Козюберда	1. Горбенко
2. Дмитро	2. Іван
3. Олександрович	3. Дмитрович
4. Харківський національний університет імені В. Н. Каразіна, Харків, 61022, Україна	4. Харківський національний університет імені В. Н. Каразіна, Харків, 61022, Україна
5. Факультет комп'ютерних наук	5. Факультет комп'ютерних наук
6. Курс 5 (1 курс магістратури)	6. Доктор технічних наук
7. Місце проживання: вул. Академіка Вальтера, 14	7. Професор
8. Телефон: +380992280147	8. Телефон: +380503239026
9. E-mail: dima2309lol@gmail.com	9. E-mail: Gorbenkol@iit.kharkov.ua

Науковий керівник Горбенко І.Д.

Автор роботи Козюберда Д.О.

Рішенням конкурсної комісії Харківського національного університету імені В.Н. Каразіна студент Козюберда Д.О. рекомендується для участі у II турі Всеукраїнського конкурсу студентських наукових робіт із напрямку «Інформатика і кібернетика» у номінації «Кібернетика: Інформаційні системи та технології»

Голова конкурсної комісії Рассомахін С.Г.

21 01 2022 року

Рисунок В.1 – Відомості про наукову публікацію

АНОТАЦІЯ

ШИФР РОБОТИ

67201GOGVM

НАЗВА РОБОТИ

«СИСТЕМА ТА ЗАСОБИ ЕЛЕКТРОННОГО ГОЛОСУВАННЯ НА ОСНОВІ ТЕХНОЛОГІЇ БЛОКЧЕЙН»

Мета роботи полягає в дослідженні системи та засобів електронного голосування на основі технології Блокчейн.

Основними методами досліджень є аналіз обраних технологій.

Завдання, яке вирішується у даній роботі полягає в тому, щоб обрати та розглянути перспективну модель електронного підпису на основі багатовимірного перетворення, що може бути використана у контексті електронного голосування та безпосередньо розгляд моделі електронного голосування на основі Блокчейн.

Робота є актуальною, оскільки її результати можуть бути використані у подальшому дослідженні чи реалізації напрямку електронного голосування з використанням Блокчейн технологій та електронного підпису на основі багатовимірних перетворень чи в інших наукових роботах. А також для обґрунтування та вибору вимог до інших перспективних систем електронного підпису, вдосконалення розглянутих моделей електронного підпису та електронного голосування, тощо.

Наукова робота містить 30 сторінок, 19 рисунків, 10 таблиць, 14 посилань на джерела, 1 додаток.

Ключові слова: ІНФОРМАЦІЙНА БЕЗПЕКА, ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ, ЕЛЕКТРОННИЙ ПІДПИС, ЕЛЕКТРОННЕ ГОЛОСУВАННЯ, БЛОКЧЕЙН, БАГАТОВИМІРНА КРИПТОГРАФІЯ, МАТЕМАТИЧНА МОДЕЛЬ, ПРОГРАМНА МОДЕЛЬ.

Рисунок В.2 – Анотація наукової публікації