Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Програмної інженерії
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка
другий (магістерський)
(рівень вищої освіти)

Дослідження сучасних алгоритмів цифрового підпису та їх практична реалізація
(тема)

Виконала: студентка 2 курсу, групи ІПЗм–17–2
спеціальності 121–Інженерія програмного забезпечення
(код і повна назва спеціальності)
Освітньо–професійної програми
Інженерія програмного забезпечення
(повна назва освітньої програми)
Кривко Є.С.
(прізвище, ініціали)
Керівник проф. Качко О.Г.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри, проф. _____ З.В.Дудар

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121–Інженерія програмного забезпечення

(код і повна назва)

Освітньо–професійна програма Інженерія програмного забезпечення

(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентці Кривко Євгенії Сергіївні

(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження сучасних алгоритмів цифрового підпису та їх практична реалізація

затверджена наказом по університету від " 18 " квітня 2019 р № 546С

заповнюється вручну після отримання наказу

2. Термін подання студентом роботи до екзаменаційної комісії _____

3. Вихідні дані до роботи реалізація постквантового алгоритму Dilithium, пояснювальна записка. Використовувати ОС Windows, середовище розробки Visual Studio Code, мову програмування Golang

4. Перелік питань, що потрібно опрацювати в роботі мета роботи, аналіз проблемної галузі і постановка задачі, дослідження сучасних алгоритмів цифрового підпису з точки зору їх захищеності від пост-квантових алгоритмів злому, дослідження математичного аппарату, що дозволяє подолати ці вразливості, аналіз пост-квантових алгоритмів цифрового підпису, реалізація найкращого пост-квантового алгоритму

Зворотний бік бланку завдання

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Мета завдання, постановка задачі, методи й алгоритми, схема реалізованого алгоритму, приклади програми, опис отриманих результатів, UML діаграми,

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка*
1.	Аналіз предметної галузі	13.02.2019	Виконано
2.	Огляд сучасних та пост-квантових алгоритмів цифрового підпису	20.02.2019	Виконано
3.	Аналіз алгритму «Dilithium» та його реалізація	15.03.2019	Виконано
4.	Підготовка пояснювальної записки	10.04.2019	Виконано
5.	Спецчастина	15.05.2019	Виконано
6.	Підготовка презентації та доповіді	20.05.2019	Виконано
7.	Попередній захист	05.06.2019	Виконано
8.	Нормоконтроль, рецензування		
9.	Занесення диплома в електронний архів		
10.	Допуск до захисту у зав. кафедри		
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання 07 лютого 2019 р.

Студент Кривко Є.С.
(підпис)

Керівник роботи _____ проф. Качко О.Г.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до роботи містить: 95 с., 5 рис., 13 форм., 8 табл., 24 джерела.

ЕЛЕКТРОННИЙ ЦИФРОВИЙ ПІДПИС, КВАНТОВИЙ КОМП’ЮТЕР, ПОСТ-КВАНТВИЙ АЛГОРИТМ, МАТЕМАТИЧНА МОДЕЛЬ, АЛГОРИТМ ШОРА, АНАЛІЗ, ПОРІВНЯЛЬНИЙ АНАЛІЗ, DILITHIUM.

Об’єктом дослідження є сучасні алгоритми цифрового підпису, вразливі та стійкі до атак квантових компьютерів.

Метою роботи є створення класифікації та проведення порівняльного аналізу сучасних алгоритмів цифрового підпису, що будуть стійкими у постквантовий період. А також реалізація найкращого з таких алгоритмів. Методи розробки базуються на мові програмування Golang та його бібліотек.

У результаті роботи проаналізовано проблему сучасних поширених алгоритмів у постквантовий період, досліджено математичні моделі, що вирішують питання вразливості до пост-квантових атак, проаналізовано посквантові алгоритми, обрано найкращий з них та реалізовано його.

DIGITAL SIGNATURE, QUANTUM COMPUTER, POST-QUANTUM ALGORITHM, MATHEMATICAL MODEL, SHOR'S ALGORITHM, ANALYSIS, COMPARATIVE ANALYSIS, CRYPTOGRAPHY, DILITHIUM.

The object of the research is modern digital signature algorithms that are vulnerable and resistant to quantum computer attacks.

The aim of this work is the creation of a classification and a comparative analysis of modern algorithms of digital signatures that will be sustainable in pasttwenty period.

As well as the implementation of the best of such algorithms. Development methods are based on Golang programming language and its libraries.

As a result of work analyzes the problem of modern distributed algorithms in post-quantum time, investigated mathematical models that cover the issues of vulnerability to post-quantum attacks, analyzed post-quantum algorithms, selected the best of them and implemented it.

ЗМІСТ

ВСТУП

За останні три десятиліття криптографія з відкритим ключем стала незамінним компонентом нашої глобальної комунікаційної цифрової інфраструктури. Цифрові комунікаційні мережі підтримують безліч програм, важливих для нашої економіки, нашої безпеки та таких сфер життя, як мобільні телефони, інтернет-торгівля, соціальні мережі та хмарні обчислення. У світі, де майже всі ключові сфері життя пов’язані з цифровими технологіями так комунікаціями, можливість людей, бізнесу та урядів безпечно спілкуватися є надзвичайно важливою.

Багато з найважливіших комунікаційних протоколів головним чином працюють на трьох основних криптографічних функціях: шифрування відкритим ключем, цифрові підписи та обмін ключами. В даний час ці функціональні можливості в першу чергу реалізуються за допомогою обміну ключами Діффі-Хеллмана, криптосистеми RSA (Rivest-Shamir-Adleman) і криптосистем еліптичної кривої. Безпека їх залежить від рівня складності деяких теоретичних задач, таких як цілочисельна факторизація або проблема обчислення дискретного логарифмування над різними групами.

У 1994 році Пітер Шор з Bell Laboratories показав, що квантові комп’ютери, нова технологія, що використовує фізичні властивості речовини і енергії для виконання розрахунків, може ефективно вирішити кожну з цих проблем, тим самим визначаючи всі криптосистеми відкритих ключів, що працюють базуючись на вищезгаданих алгоритмах, як ненадійні та неможливі до використання в майбутньому [1]. Таким чином, коли буде винайдений досить потужний квантовий комп’ютер, то багато форм сучасного зв’язку – від обміну ключами до шифрування з метою цифрової аутентифікації – опиняться в небезпеці.

Відкриття того, що квантові комп’ютери можна використовувати для вирішення певних проблем швидше, ніж класичні комп’ютери, викликало великий інтерес до квантових обчислень, тому останнім часом спостерігається значна

кількість досліджень у сфері квантових комп’ютерів – машин, що використовують квантово-механічні явища для вирішення математичних проблем, які важко або неможливо вирішти за допомогою звичайних комп’ютерів.

Коли будуть побудовані великомасштабні квантові комп’ютери, вони зможуть зламати багато криптосистем з відкритимии ключами, що зараз використовуються у більшості випадків. Це серьйозно вплине на конфіденційність та цілісність цифрових комунікацій в інтернеті. Мета пост-квантової криптографії, що також назіваєється квантово-стійкою криптографією, полягає в розробці криптографічних систем, захищених як від квантових, так і від классичних комп’ютерів, а також може бути інтегрована в протоколи та мережі, що вже існують та використовуються.

Для розгортання сучасної системи криптографії знадобилося майже два десятиліття. Тому, незалежно від того, що зараз неможливо оцінити точний час початку ери квантових обчислень, необхідно вже зараз працювати над алгоритмами, стійкими до атак квантових комп’ютерів. Саме тому дослідження сучасних алгоритмів цифрового підпису та їх практична реалізація є актуальним й дуже важливим напрямком в наш час.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз досягнень у сфері квантових комп’ютерів

Дослідження доцільності побудови великомасштабних квантових комп’ютерів почали серйозно проводитися після відкриття Петром Шором 1994 року поліномно-часового квантового алгоритму для цілочисельної факторизації [1]. Тоді ще не було зрозуміло, чи буде квантовий компьютер коли-небудь масштабований достатньо для виконання алгоритму Шора. Багато провідних експертів допускали, що квантові стани занадто крихкі та кількість помилок занадто значна для великомасштабного квантового обчислення, яке коли-небудь це буде досягнуто. Дана ситуація змінилася наприкінці 1990-х років з розвитком квантових кодів корекції помилок і порогових теорем [2]. Ці порогові теореми показують, що якщо частота помилок на одну логічну операцію («квантові ворота») в квантовому комп’ютері може бути доведена нижче фіксованого порогу, то довільно довгі квантові обчислення можуть бути здійснені надійним і стійким до відмов способом через включення похибки кроку корекції протягом виконання квантового обчислення [3].

Протягом багатьох років експериментатори поступово розробляли вдосконалене обладнання з більш низькими показниками. Одночасно теоретики розробили нові квантові процедури корекції помилок, що дають вищі пороги стійкості до відмов. Останнім часом деякі експерименти із застосуванням іонних пасток та надпровідних ланцюгів продемонстрували універсальні набори квантових затворів, які номінально нижче найвищих теоретичних порогів стійкості до відмов (близько 1%) [4, 5]. Це важливий етап, який стимулював збільшення інвестицій як з боку уряду, так і з боку промисловості. Проте, очевидно, що для переходу від сучасних лабораторних демонстрацій із залученням декількох кубітів до великомасштабних квантових комп’ютерів, що включають тисячі логічних кубітів, закодованих, можливо, в сотні тисяч або мільйони фізичних кубітів, необхідні значні довгострокові зусилля.

Паралельно з розробкою цифрових квантових комп’ютерів загального призначення було прикладено зусилля з розробки аналогових квантових комп’ютерів спеціального призначення, таких як квантові відпалювачі (наприклад, машина D-Wave), аналогові квантові імітатори та пристрої для вибірки бозонів. Деякі з цих пристроїв були розширені до більшої кількості кубітів, ніж у цифрових квантових комп’ютерів. Однак, внаслідок їх спеціалізованої природи, ці аналогові квантові пристрої не вважаються важливими для криптоаналізу.

21 грудня 2018 року Дональд Трамп підписав «Закон про Національну квантову ініціативу (NQI)» [6], який передбачає вилучення протягом наступних п’яти років 1,2 млрд доларів на дослідження в області квантових комп’ютерів, що свідчить про серьйозність та актуальність проблеми. На виставці CES 2019 компанія IBM представила свій перший комп’ютерний комп’ютер IBM Q System One, який було створено для комерційного використання.

Оскільки класичні комп’ютери об’єднують декілька компонентів в інтегровану архітектуру, оптимізовану для спільної роботи, IBM застосував такий же підхід до квантових обчислень з першою повністю інтегрованою універсальною квантовою обчислювальною системою.

Квантовий комп'ютер IBM Q System One включає в себе систему з 20 кубітів. Залізо здатне до самокалібровки і оптимізовано для роботи в кріогенних умовах, ці заходи необхідні для зменшення числа помилок. Крім того, комп'ютер має власну високопродуктивну кріогенну систему, а за заявою IBM допустимо проводити діагностику, обслуговування і навіть модернізацію системи без її виключення, зі збереженням можливості роботи користувачів.

Важливою характеристикою, на якій виробник акцентує увагу, є вбудований функціонал для підключення квантового комп'ютера до хмарної системі. Таким чином відбувається функціональний перехід від спеціалізованих квантових систем на чіпі, використовуваних в експериментальних цілях, до повноцінної інтеграції квантових обчислень для призначених для користувача систем і потреб бізнесу.

З метою розвитку екосистеми квантових обчислень IBM так само оголосила про створення консорціуму з ExxonMobil, Європейською організацією з ядерних

досліджень (ЦЕРН) і Fermilab, для пошуку максимального числа завдань, для яких застосування подібних інтегрованих квантових систем було б ефективним. Дана група організацій не є закритою і вже оголошено запрошення до співпраці для інших зацікавлених сторін.

У другій половині 2019 року IBM відкриє IBM Q Quantum Computation Center, розташований у Нью-Йорці, для розширення комерційної комп'ютерної програми IBM, яка вже включає системи дослідницького центру Thomas J. Watson в Йорктауні, Нью-Йорк. У цьому новому центрі будуть розташовані деякі з найдосконаліших у світі хмарних квантових обчислювальних систем, які будуть доступні для членів мережі IBM Q, світової спільноти провідних компаній зі списку Fortune 500, стартапів, академічних установ і національних дослідницьких лабораторій IBM для просування квантових обчислень і вивчення практичних додатків для бізнесу та науки. Винайдення квантового комп'ютеру, що буде здатним зламати сучасні найрозповсюдженіші алгоритми шифрування, стає все більш ймовірним та близьким майбутнім.

Наразі не можна точно сказати, коли будуть доступні масштабовані в достатній мірі квантові комп'ютери, але цілком ймовірно, що квантовий комп'ютер, здатний в лічині години атакувати RSA-2048, може бути побудований до 2030 р., а це вже серйозна довгострокова загроза для криптосистеми, що в даний час стандартизована в NIST.

1.2 Аналіз можливостей квантових обчислень для криптоаналізу сучасних криптосистем

Однією з потенційних можливостей використання квантових обчислень є злам відкритого ключа криптографічних шифрів, наприклад, RSA, що зараз використовуються в усьому світі для захисту багатьох транзакцій в Інтернеті та іншмх мережах комп'ютерної комунікації. Це пояснюється тим, що алгоритми з

відкритим ключем засновані на факторингу великого числа на два прості числа. Для виконання цього розрахунку на класичному комп’ютері для 2048-бітного ключа, що зазвичай використовуються в наш час, знадобляться мільярди та мільярди років. Але математик Пітер Шор винайшов квантовий алгоритм, який може виконати ці обчислення досить швидко на квантоввому комп’ютері зі значною кількістю кубіт. Сьогодні ще не існує квантових комп’ютерів з необхідно кількістю кубіт.

Варто зауважити, що алгоритми симетричних ключів, таких як AES, не засновані на факторизації великого числа і не можуть бути атаковані з використанням алгоритма Шор. Хоча квантовий комп’ютер може злегка прискорити роботу атаку «грубої сили» за допомогою алгоритму гловера, це не є дієвим, тому що це легко компенсувати збільшивши розмір ключа вдвічі. Так, наприклад, складність злому коду AES 256-бітним ключем на квантовому комп’ютері були б настільки ж складними, як злом з 128-бітовим ключем на класичному комп’ютері. До основних задач, які можуть бути вирішені на квантовому комп’ютері, в першу чергу необхідно віднести такі:

– квантовий алгоритм факторизації Шора;

– квантовий алгоритм Гровера;

– квантовий алгоритм Шора вирішення дискретного логарифму в скінченному полі;

– квантовий алгоритм Шора вирішення дискретного логарифму в групі точок еліптичної кривої;

– квантовий алгоритм криптоаналізу для перетворень в фактор кільці.

1.3 Криптографічні системи, вразливі до атак квантових комп'ютерів

На сьогодні вже існують квантові алгоритми, які дають змогу проводити атаки на такі асиметричні криптосистеми:

– системи, що базуються на складності факторизації великого цілого числа (RSA)];

– системи, що базуються на складності вирішення дискретного логарифму в скінченному полі Галуа (DSA);

– системи, що базуються на складності вирішення дискретного логарифму в групі точок еліптичної кривої (ECC);

– системи на базі решіток (NTRU).

Усі вказані криптосистеми відносяться до класу ймовірно-стійких. А ця ймовірна стійкість як раз і визначається можливостями появи квантових комп’юте рів, і, як наслідок, вирішення задачі повного розкриття. Існують також квантові алгоритми, що можуть використовуватися для проведення криптоаналізу симетричних криптосистем, в першу чергу блокових та потокових симетричних шифрів.

Забезпечення необхідного рівня стійкості симетричних криптографічних систем при появі квантового комп'ютера можна тільки за допомогою збільшення розмірів довжини блока та довжини ключа. Але і за умови збільшення розмірів системних параметрів симетричні системи можуть бути при певних параметрах уразливими для квантового криптоаналізу.

Складність вирішення задач факторизації в кільцях та дискретного логарифмування в скінченному полі при застосуванні класичних комп'ютерів носить в кращому випадку субекспоненційний характер. Поява нових математичних методів факторизації та дискретного логарифмування та їх практична реалізація можуть привести до того, що перетворення в кільці та в скінченному полі будуть не стійкими до атак типу «повне розкриття» [7]. Найпопулярніші алгоритми цифрового підпису, що поширені сьогодні, та вплив велико-масштабованих комп’ютерів на ці алгоритми представлено у таблиці 1.1.

Квантовий алгоритм Шора у загальному випадку має поліноміальну складність вирішення задач криптоаналізу, наприклад факторизації чи дискретного логарифмування. При його застосуванні складність дискретного логарифмування складає час $O(n^3)$ та використання $O(n)$ кубітів, де n – порядок

базової точки. Квантовий алгоритм Шора у загальному випадку має поліноміальну складність вирішення задач криптоаналізу, наприклад факторизації чи дискретного логарифмування. При його застосуванні складність дискретного логарифмування складає час $O(n^3)$ та використання $O(n)$ кубітів, де n – порядок базової точки.

Таблиця 1.1 – Вплив квантових комп’ютерів на поширені алгоритми цифрового підпису

Алгоритм шифрування	Тип	Призначення алгоритму	Вплив велико-масштабованих квантових комп’ютерів
AES	Симетричний ключ	Шифрування	Необхідно збільшити розмір ключа
SHA-2, SHA-3	-	Хеш-функція	Необхідний збільшити розмір вхідного повідомлення
RSA	Відкритий ключ	Цифровий підпис	Ненадійний
ECDSA, ECDH (еліптичні криві)	Відкритий ключ	Цифровий підпис	Ненадійний
DSA (скінчене поле, поле Галуа)	Відкритий ключ	Цифровий підпис	Ненадійний

Єдине на чому може базуватися стійкість асиметричних криптосистем при квантовому крипто аналізі – це те, що в найближчі роки не з'являться квантові комп'ютері з необхідним для цього числом кубітів.

1.4 Аналіз стану пост-квантової криптографії

Найбільш важливим напрямком криптографії на сьогодні є використання відкритих ключів для цифрових підписів і створення ключів. Як згадувалося на

початку розділу 1, побудова великомасштабного квантового комп'ютера зробила б багато з цих криптосистем небезпечними. Зокрема, це стосується труднощів цілочисельної факторизації, таких як RSA, а також тих, які ґрунтуються на складності проблеми дискретного логарифмування. І навпаки, вплив на симетричні ключові системи не буде настільки різким (див. таблицю 3). Алгоритм Гровера забезпечує квадратичне прискорення алгоритмів квантового пошуку порівняно з алгоритмами пошуку на класичних комп'ютерах, тому подвоєння розміру ключа буде достатньо для збереження безпеки. Так як експоненційна швидкість для алгоритмів пошуку неможлива, це дозволяє припустити, що симетричні алгоритми і хеш-функції можуть використовуватися в квантовій ері.

Отже, пошук алгоритмів, які вважаються стійкими до атак з боку як класичних, так і квантових комп'ютерів, зосереджений на алгоритмах відкритих ключів. Серед алгоритмів, що будуть стійкими до квантових атак, є такі, що ґрунтуються на решітках, кодах і багатовимірних поліномах, а також на декількох інших. Дані пост-квантові примітиви будуть розглянуть у наступних розділах.

Постквантовая криптографія пропонує нові алгоритми шифрування, які є стійкими до злому як «класичними», так і квантовими комп'ютерами. На даний момент у сфері постквантової криптографії ведуться наукові дослідження за пошуком таких алгоритмів. Однак до сих пір на ринку відсутні єдині визнаних стандартів постквантового шифрування, які довели свою ефективність. Можлива ситуація, при якій з розвитком квантових комп'ютерів обраний алгоритм вже не забезпечить необхідний рівень захисту, і його доведеться міняти. Тому важливо вибрати оптимальний алгоритм відразу. Інша особливість постквантовой криптографії - це високі вимоги до обчислювальних ресурсів. З одного боку, дані алгоритми можна реалізувати практично в будь-якому програмному коді, з іншого боку – при зростанні обсягів даних, що захищаються потужності наявного обладнання може виявитися недостатньо.

У 2016 році Національний інститут стандартів та технологій (далі NIST) США з ініціював процес пошуку, оцінки та стандартизації одного або більше

криптографічних алгоритмів із відкритим ключем [8]. Проект має назву - Post-Quantum Cryptography Standardization.

Передбачається, що нові криптографічні стандарти публічного ключа визначатимуть один або декілька додаткових некласифікованих, публічно розкритих цифрових підписів, шифрування з відкритим ключем та алгоритми встановлення ключів, які доступні у всьому світі, і здатні захистити важливу інформацію в майбутньому, в тому числі і з появою квантових комп'ютерів.

Спочатку були зібрані усі можливі публічні зауваження щодо мінімальних вимог до пост-кантових алгоритмів, далі були відібрані кандидати, що пройшли до першого раунду оцінки. Наприкінці січня 2019-го року оголосили результати й список алгоритмів, що пройшли до наступного раунду. У кандидатів було декілька місяців, щоб вдосконалити свої алгоритми, наприклад, покращити продуктивність, але без суттєвих змін, і вже 15-го березня 2019 року почався другий раунд оцінки алгоритмів. NIST планує до 2022 року визначити нові стандарти й алгоритми, що будуть стійкими в єру квантових комп’ютерів.

Майже одночасно с NIST роботу над пошуком пост-квантових алгоритмів шифрування та розбкою нових постквантових стандартів почав Європейський інститут телекомунікаційних стандартів. В даному інституті було сформовано новий напрямок «Квантово-захищена криптографія», котрий займається питанням безпеки в постквантовий період, й повинен за результатами досліджень прийняти группу стандартів для постквантового періоду. На данний момент вже готові попередні версії стандартів.

1.5 Постановка задачі

В межах магістерської роботи буде проведено аналіз сучасних алгоритмів цифрового підпису, проблеми безпеки, пов’язанні з розвитком квантових комп’ютерів, та постквантові алгоритми цифрового підпису, що можуть розв’язати

ці проблеми. Так як ще немає офіційно встановлених стандартів, необхідно проаналізувати направлення постквантової криптографії, та алгоритми, що є потенційно стійкими до злому, на особливості використання, стійкість до злому відомими атаками на классичних та квантових комп’ютерах. Необхідно обрати алгоритм, що буде найкращим в порівняльному аналізі, провести більш детальні дослідження й реалізувати в можливому варінті. Так як ще не існує квантових комп’ютерів, які б представляли реальну загрозу, то доказ безпеки алгоритмів може бути лише теоретичним.

Метою магістерської роботи є реалізація постквантового алгоритму цифрового підпису, так як наразі існує досить мало реалізацій алгоритмів цифрових підписів, стійких до атак квантових комп’ютерів.

Отже, для реалізації стійкого до квантових атак алгоримту цифрового підпису нехідно виконати наступі задачі:

– проаналізувати прогрес досліджень у сфері квантових комп'ютерів з метою оцінювання реальної загрози сучасним алгоритмам цифрових підписів;

– дослідити сучасні алгоритми цифрового підпису з точки зору їх захищеності від атак квантових комп'ютерів;

– дослідити математичний апарат, що дозволяє вирішити вразливості до атак квантових комп'ютерів;

– дослідити постквантові-алгоритми, що наразі вирішують проблему атаки квантовими комп'ютерами;

– проаналізувати й реалізувати найперспективніший з пост-квантових алгоритмів;

– оцінити отримані результати.

Для реалізації алгоритму було обрано мову Golang та середовище розробки Visual Studio Code. Дана мова програмування є досить поширеною, має усі необхідні для реалізації математичні та криптографічні бібліотеки, може бути інтегрована з будь-якою іншою мовою програмування й має гарні показники продуктивності роботи, тому реалізація на даній мові програмування буде актуальною.

2 АНАЛІЗ ПОСТКВАНТОВИХ АЛГОРИТМІВ ЦИФРОВОГО ПІДПИСУ

2.1 Квантовий алгоритм факторизації Шора

В наш час широке розповсюдження та застосування знайшов алгоритм асиметричного перетворення в кільці, який отримав назву RSA перетворення. За складністю основним етапом його криптоаналізу, тобто знаходження особистого ключа, є факторизації модуля перетворення N. Усі відомі нині класичні алгоритми факторизації мають або експоненційну, або субекспоненційну складність. Вважається, що найкращим з точки зору мінімізації складності факторизації – є алгоритм загального решета числового поля, або його модифікації. Але застосування алгоритму загального або спеціального решета числового поля для реальних значень загальних параметрів, наприклад $N \geq 2^{2048}$, не можуть бути реалізовані. Часова складність таких алгоритмів оцінюється як субекспоненційна, наприклад у вигляді $O(\exp((c + o(1))(\ln n)^{1/3}(\ln \ln n)^{2/3}))$.

В той же час квантовий алгоритм має поліноміальну складність. Він здатен розкласти складене число на прості множники приблизно за такий час O(n3) та з використанням O(n) кубітів [1]. Розглянемо його детальніше. Поліноміальний за часом алгоритм Шора, запропонований в 1994 році, дозволяє факторизувати N - значне число на квантовому комп'ютері з певною ймовірністю та з обмеженою помилкою. Пропозиція та оцінки складності алгоритму Шора вразили, пробудивши широкий інтерес до квантових обчислень. В алгоритмі Шора використано ефект квантового паралелізму, який запропоновано для отримання суперпозиції всіх значень функції за один крок. Після цього здійснюють квантове перетворення Фур'є функції. Подальші обчислення дозволяють визначити з великою ймовірністю період N, який використовується для його факторизації. З точки зору складності, найбільшу складність становить перетворення Фур'є, що базується на алгоритмі швидкого перетворення Фур'є.

Шор також показав, що його алгоритм дозволяє розкласти число N з часовою складністю $O(\log^2 N \log^3(\log N))$ з використанням O(log N) логічних кубітів [9].

Таким чином, значимість алгоритму полягає в тому, що при використанні квантового комп'ютера з декількома сотнями логічних кубітів він дозволяє, наприклад, зламати RSA криптоперетворення, роз клавши модуль перетворення N, тобто знайти множники модуля N. Уже при $N \geq 2^{1024}$ це зробити практично неможливо, якщо використовувати відомі класичні алгоритми.

Попередні оцінки показують, що алгоритм Шора дозволить вирішити проблему факторизації числа N за допомогою вирішення еквівалентної проблеми – для певного числа a, що є взаємно простим з N, знайти порядок r елемента a mod N. При цьому ціле число a рекомендується вибирати випадково і рівно ймовірно. Далі, використовуючи Алгоритм Евкліда, можна визначити, чи a взаємно просте з N. Якщо a не є взаємно простим з N, то буде знайдено дільник числа N. В іншому випадку, порядок r елемента a mod N буде дільником числа N. Порівняльний аналіз класичного алгоритму і квантового алгоритмів факторизації наведено у таблиці 2.

Таблиця 2.1 – Алгоритм факторизації RSA

Алгоритм факторизації RSA			
Розмір модуля, бітів	Кількість необхіних кубітів, 2n	Складність квантового алгоритму	Складність класичного алгоритму
512	1024	$0{,}54*10^{9}$	$1{,}6*10^{19}$
768	1536	$1{,}8*10^{9}$	$9{,}9*10^{22}$
1024	2048	$4{,}3*10^{9}$	$1{,}2*10^{26}$
2048	4096	$34*10^{9}$	$1{,}35*10^{35}$
3072	6144	$12*10^{10}$	$5*10^{41}$
15360	30720	$1{,}5*10^{13}$	$9{,}2*10^{80}$

Аналіз даних таблиці 1 показує, що для зламу RSA криптосистеми з розміром модуля у 15360 бітів (розмір головного сертифікату відкритого ключа США), необхідно лише $1.5 \cdot 10^{13}$ операцій на квантовому комп'ютері, тоді як класична обчислювальна система повинна виконати порядку 10^{80} операцій. Тобто RSA система буде зламана за поліноміальний час. Крім того, як слідує із таблиці 2.1, навіть суттєве збільшення модуля, наприклад 2^{3072} та більше, не врятує RSA криптосистему від злому.

2.2 Квантовий алгоритм Шора дискретного логарифму в скінченому полі та групі точок еліптичної кривої

Проведений аналіз показує [1], що алгоритм Шора дискретного логарифмування в групі точок еліптичної кривої та в скінченному полі має однакові кроки, єдине в чому є різниця – це представлення точок еліптичної кривої та елементів поля.

Розглянемо їх криптографічну стійкість та зробимо відповідні оцінки для них. Нині вважається, що вирішення задач дискретного логарифмування в групі точок еліптичних кривих найбільш ефективно може бути реалізованим з використанням ρ- та λ- методів Полларда. Для них складність можна оцінити як: O(q), де q - кількість точок еліптичної кривої. В той же час квантовий алгоритм Шора у загально му випадку має поліноміальну складність вирішення такого класу задач, як, наприклад, факторизації. Отже, алгоритм Шора на квантових комп’ютерах здатен розв’язати логарифмічне рівняння приблизно за такий час O(n3) та з використанням O(n) кубітів, де n - розмір базової точки. Збільшення розміру порядку базової точки при криптоаналізі з використанням квантового алгоритму не дає суттєвого збільшення криптографічної стійкості криптографічної системи на еліптичних кривих.

При збільшені модуля, складність дискретного логарифмування класичними методами в групі точок еліптичної кривої зі збільшенням порядку базової точки збільшується суттєво. Але для квантового алгоритму проблемою є велика кількість кубітів, яка необхідна для проведення квантової атаки.

На сьогодні «рекордом класичного криптоаналізу» в групі точок еліптичних кривих є рішення дискретного логарифмічного рівняння в полі GF(2109), при розмірі базової точки 108 бітів. Така задача була розв'язана у 2000 році французькими криптологами, для її вирішення знадобилося 9500 комп’ютерів та трохи більше 4 місяців.

Також нині існує багато алгоритмів та методів, за допомогою яких можна знайти розв'язок дискретного логарифму в скінченному полі. Найкращим класичним алгоритмом дискретного логарифмування в скінченному полі є метод решета числового поля. Для нього складність можна оцінити як: $O(\exp(3^{3/2}(\ln(P)\ln(\ln(P)))^{1/3})$, але, класичні методи криптоаналізу систем мають субекспоненційну або експоненційну складність.

В той же час алгоритм Шора, маючи поліноміальну складність, дозволяє вирішити проблему дискретного логарифму в полі з суттєво меншою складністю [10]. Порівняльний аналіз складності алгоритму решета числового поля та алгоритму Шора дискретного логарифмування в скінченному полі [7] наведено в таблиці 2.2.

Таблиця 2.2 – Алгоритм розв’язку дискретного логарифмічного рівняння

Алгоритм розв’язку дискретного логарифмічного рівняння			
Розмір модуля перетворення, бітів	Кількість необхіних кубітів, ~3n	Складність квантового алгоритму	Складність класичного алгоритму
512	1536	$1,3*10^{8}$	$4,9*10^{15}$
768	3072	$0,1*10^{10}$	$3,3*10^{20}$
1024	6144	$0,9*10^{10}$	$5,3*10^{26}$
2048	9216	$2,9*10^{10}$	$1,4*10^{31}$
3072	24486	$0,5*10^{12}$	$8,2*10^{44}$
15360	46080	$3,6*10^{12}$	$5,9*10^{56}$

Із таблиці 2.2 видно, що збільшення розміру модуля перетворення і, відповідно, особистого ключа, не забезпечує необхідного збільшення складності дискретного логарифмування в скінченному полі, наприклад, при зломі електронного цифрового підпису чи направленого шифрування.

Для модуля $P\geq 2^{3072}$ складність дискретного логарифмування в скінченному полі складає $1.4\cdot10^{31}$, а з застосуванням алгоритму Шора всього $2.9\cdot10^{10}$ операцій[11]. Але для квантового алгоритму залишається велика кількість кубітів, яка необхідна для проведення квантової атаки. Скоріше всього вона тривалий час буде недоступною.

2.3 Вимоги до постквантових алгоритмів цифрового підпису

Не всі протоколи безпеки та криптографічні алгоритми вразливі до квантових атак; деякі з них вважаються безпечними, тоді як деякі, як вже відомо, вразливі. Контроль безпеки, який вважається квантово-безпечним сьогодні, з часом і при достатніх дослідженнях може бути визначений вразливим. Без доказів того, що алгоритм вразливий до квантової атаки, криптографічний примітив та протоколи, які його використовують, вважаються квантово-безпечними, якщо вони добре вивчені та протистоять атакам, з використанням усіх відомих квантових алгоритмів.

Електронний підпис (ЕП) дозволяє підтвердити авторство електронного документа (будь то реальна особа або, наприклад, аккаунт в кріптовалютной системі). Підпис пов'язаний як з автором, так і з самим документом за допомогою криптографічних методів, і не може бути підробленим за допомогою звичайного копіювання. ЕЦП - це реквізит електронного документа, отриманий в результаті криптографічного перетворення інформації з використанням закритого ключа підпису, що дозволяє перевірити відсутність спотворення інформації в електронному документі з моменту формування підпису (цілісність), приналежність підпису власникові сертифіката ключа підпису (авторство), а в разі успішної перевірки підтвердити факт підписання електронного документа (неспростовності). Надійність цифрового підпису визначається стійкістю до криптоаналітичних атак двох її компонент: хеш-функції і самого алгоритму ЕЦП. Стійка схема цифрового підпису повинна використовувати хеш-функцію, що володіє наступними властивостями:

– Однобічність. Нехай дано хеш-значення H (M) деякого невідомого повідомлення M. Тоді обчислювально неможливо визначити M за наявним H (M).

– Стійкість до зіткнення (колізії). Нехай дано повідомлення M і його хеш-значення H (M). Тоді обчислювально неможливо визначити M 'таке, що H (M) = H (M'). Це властивість еквівалентно властивості однобічності.

– Висока стійкість до зіткнення (колізії). Обчислювально неможливо знайти два довільних повідомлення M і M ', для яких H (M) = H (M').

Як вже зазначалося вище, надійність цифрового підпису визначається стійкістю до криптоаналітичних атак алгоритму ЕЦП, що залежать від кожного окремого алгоритму шифрування. Вимоги до постквантових алгоритмів, ще точно не задані. NIST (Національний інститут стандартів та технологій США) та ESTI (Европейский інститут стандартизації телекомунікацій) у 2016 році почали роботу над стандартизацією та пошуком алгоритмів шифрування, в тому числі і для електронного цифрового підпису. У наступних підрозділах розглянемо вимоги, що були висунуті до кандидатів програм стандаризації постквантових алгоритмів.

2.3.1 Вимоги NIST до кандидатів постквантових алгоритмів електронного підпису

Мінімальні критерії, що визначив NIST до кандидатів постквантових алгоритмів на першому єтапі відбору до першого раунду були наступні [8]: реалізація на мові програмування C, проходження тестів з відомими відповідями, реалізація в широкому діапазоні апаратних та програмних платформ. Далі алгоритми, що пройшли до першого раунду оцінювали за трьома показниками: безпека, вартість і продуктивність, й алгоритм та особливості реалізації.

Як і в минулих змаганнях між AES и SHA-3, безпека є найважливішим фактором при оцінюванні кандидатів на звання постквантового алгоритму. NIST має намір стандартизувати постквантові алгоритми з публічним (відкритим) ключем для використання в різноманітних протоколах, таких як протокол захисту транспортного рівня (TLS), «безпечна оболонка» (SSH), обмін ключами в інтернеті (IKE), безпека інтернет-протоколу (IPsec) і розширення безпеки доменних систем (DNSSEC). Схеми повинні оцінюватись за рівнем безпеки, який вони надають у заданих (та інших) додатках. Системи цифрового підпису повинні дозволяти

підписи, що не можуть бути зламані, стосовно адаптивного вибору повідомлення (модель безпеки – EUF-CMA) з умовою безпеки, що доступ зловмисника менше ніж 2^{64} ораних повідомлень.

Усвідомлюючи, що існує значна невизначеність в оцінці сильних сторін безпеки пост-квантових алгоритмів-кандидатів, NIST виділила п'ять категорій безпеки, щоб краще порівнювати рівень безпеки, наведений у поданнях. Відправників попросили надати попередню класифікацію згідно з визначеннями, наданими у Federal Register Notice [12].

NIST також згадував про інші властивості безпеки, такі як ідеальна пряма секретність, опір бічних і багатоканальних атак, а також стійкість до зловживань. Крім того, NIST вимагала подання пакетів для узагальнення відомих криптоаналітичних атак на алгоритм та оцінки складності цих атак.

Для криптографії електронного підпису з відкритим ключем були висунуті наступні вимоги безпеки [13]:

– заміна стандарту цифрового підпису FIPS 186;

– заміна стандартів розподілу ключів SP 800-56A, SP 800-56B;

– використаня нового стандарту в протоколах TLS, SSH, IPsec, DNSSEC.

Також були висунуті наступні вимоги до стійкості алгоритмів шифрвання:

– 128 біт класичної безпеки / 64 біт квантової захищеності (запас стійкості AES-128);

– 128 біт класичної безпеки / 80 біт квантової захищеності (запас стійкості SHA-256/ SHA3-256);

– 192 біт класичної безпеки / 96 біт квантової захищеності (запас стійкості AES-192);

– 192 біт класичної безпеки / 128 біт квантової захищеності (запас стійкості SHA-384/ SHA3-384);

– 256 біт класичної безпеки / 128 біт квантової захищеності (запас стійкості AES-256).

NIST визначив вартість як другий найважливіший критерій при оцінці алгоритмів-кандидатів [11]. У цьому випадку під вартістю мається на увазі

обчислювальна ефективність і вимоги до пам'яті. Це включає, наприклад, розмір відкритих ключів, зашифрованого тексту та підписів, ефективність обчислень генерації ключів, а також операцій з відкритими та приватними ключами, імовірність відмов дешифрування.

Обчислювальна ефективність по суті відноситься до швидкості алгоритму. NIST сподівається, що алгоритми-кандидати пропонуватимуть порівнянну або поліпшену продуктивність у порівнянні зі стандартними алгоритмами з публічним ключем. Вимоги до пам'яті відносяться як до вимог до розміру коду, так і до оперативної пам'яті (RAM), необхідних для реалізації програмного забезпечення, а також для кількості воріт для реалізації апаратних засобів. [11] вимагав, щоб усі подавці включали оцінки продуктивності алгоритмів, запущених на NIST PQC Reference Platform, Intel x64, Windows or Linux, та компілятор GCC. NIST провів попередній аналіз ефективності на еталонній платформі, але також запропонував громадськості провести аналогічні тести на додаткових платформах.

Процес стандартизації NIST PQC отримав багато алгоритмів кандидатів з новими та цікавими проектами та з унікальними функціями, які не існують у поточних стандартизованих алгоритмах з публічними ключами. Алгоритам, що здатні ефективно працювати на різних платформах чи алгоритми, що використовують паралелізм або набір інструкцій для досягнення більш високої продуктивності, надається перевага Крім того, прості та зрозумілі конструкції є кращими, тому їх краще розуміти, тому вони викликають довіру проектної групи та заохочують подальший аналіз.

2.3.2 Вимоги ESTI до кандидатів постквантових алгоритмів електронного підпису

Одночасно з NIST роботу над стандартизацією почав і Європейский інститут стандартів телекомунікацій – ESTI: у кластері «Безпека» сформовано новий

напрям – «Quantum-Safe Cryptography» (Квантово захищена криптографія). ETSI розробила вимоги, котрі запроповновано висунуту до кандидатів на постквантові стандарти криптографічних перетворень. Список базових вимог до пост-квантових алгоритмів:

– проходження громадського контролю та визнання науковою спільнотою;

– надійне підтвердження криптографічної стійкості;

– актуальність моделі безпеки, якій відповідають криптопримітиви;

– стійкість криптопримітивів до існуючих, в тому числі на основі квантових алгоритмів, атак;

– можливість використання криптопримітивів в безпечних протоколах розподілу ключів;

– можливість поєднання кількох функцій безпеки криптографічних протоколів (наприклад, одночасна аутентифікація та встановлення);

– можливість кількісної оцінки та порівняння заявлених класичних та квантових рівнів безпеки;

– визначеність рекомендованихрозмірів ключів для заданих рівнів безпеки (наприклад, 80, 112 біт, 128 або 256 біт);

– стійкість проти класичних атак;

– стійкість проти «квантових» атак. Зокрема, стійкість до алгоритму Гровера та Шора;

– базування на задачах, які мають необхідну складність криптографічного аналізу. Можливе ігнорування зниження рівня складності за умови, що практична стійкість зміниться несуттєво;

– стійкість проти атак з адаптивним підбором.

Щодо вимог до ефективності, то було визначено, що постквантові алгоритми повинні використовувати рекомендовані параметри розмірів загальних параметрів та ключів відповідно до заданого рівня безпеки. Швидкодія та кількість раундів перетворень не повинні залежати від платформи реалізації. Швидкодія генерації ключів і часу, необхідного для поширення нового ключа, повинна бути в межах, допустимих для практичної задачі. А також повинні виконуватись інші практичні

вимоги, такі як стійкість до відмов та інші, якщо це розумно чи необхідно в кожній конкретній ситуації.

Також ETSI висунув вимоги до розмірів параметрів та ключів електронного підпису, що наведені у таблиці 2.3.

Таблиця 2.3 – Вимоги ETSI до розмірів параметрів та ключів ЕП

Параметр Алгоритм	Час генерації ключів (RSA sign=1)	Час підпису (RSA sign=1)	Час перевірки (RSA sign=1)	Крипто-період	Розмір відкритого ключа (біт)	Розмір закритого ключа (біт)	Розмір підпису (біт)
Підпис Меркла	200 10000 500000	1 1 2	0.2 0.2 0.2	2^{20} 2^{30} 2^{40}	368 368 368	15200 22304 29344	17024 18624 20224
GLP підпис (криптографія на решітках)	0.01	0.5	0.02		11800	1620	8950
GFS підпис (code-based)	5	2000	0.02		9437184	15000000	144
pSFLASH підпис (багатовимірна криптографія)	50	1	0.1		576992	44400	296

Висунуті NIST США та ETSI ЄС вимоги дозволяють, на наш погляд, зробити тільки попередні оцінки постквантових механізмів ЕП та приймати певні рішення відносно подальших їх досліджень з метою вибору найбільш ефективних за критеріями стійкості та техніко-економічними і техніко-експлуатаційними критеріями. Також були висунуті вимоги до практичної реалізації та розгортання постквантових алгоритмів:

– простота впровадження стандартних примітивів нефахівцями. Відносно малий обсяг (ресурс) реалізації (зокрема, можливість реалізації на FPGA та вбудованих пристроях);

– відносно малий (допустимий) обсяг необхідної пам'яті під час виконання (можливість реалізації на пристрої з обмеженими ресурсами);

– практичність розмірів ключа і підпису для передачі або зберігання в ряді платформ, включаючи пристрої з обмеженими ресурсами;

– простота інтеграції в існуючі криптографічні протоколи та системи;

– низька вартість заміни або модернізації захищених технологій;

– можливість виконання криптопримітивів для декількох функцій, наприклад для забезпечення аутентифікації, розподілу ключів тощо;

– сумісність, наприклад гнучкість, у виборі хеш-функції в схемах дерева Меркле тощо.

2.4 Дослідження математичного аппарату, що дозволяє подолати вразливість до атак квантових комп’ютерів

Окрім систем, безпека яких базується на задачі факторизації дуже великих чисел, існують також системи, що спираються на обчислювально-складні задачі та мають ряд значних відмінностей, через які їх набагато важче вирішувати. Так як складність задач цих систем базується не на цілочисельній факторизації, вони не залежать від загроз обчислень на квантових комп’ютерах, і, відповідно, важаються квантово-стійкими чи постквантовими алгоритмами. Постквантова криптографія відрізняється від квантової криптографії, суть якої полягає в розробці методів захисту комунікацій, що базуються на принципах квантової фізики.

Зараз існує п’ять напрямків, в яких розвивається постквантова криптографія. Це криптографія заснована на хещ-функціях, криптографія, заснована на кодах виправлення помилок, кпритографія, заснована на решітках, криптограія, заснована на багатовимірних квадратичних системах, та шифрування з секретним ключем [11]. Далі розглянемо кожен з цих алгортмів окремо.

2.4.1 Криптографія, заснована на перетворенні на алгебраїчних решітках

Серед усіх обчислювальних завдань, що вважаються квантово-стійкими, алгоритми, що базуються на решітках, отримали найбільшу увагу протягом останнього десятиліття. Криптографія, що заснована на решітках, є досить швидкою і вважається стійкою до квантових атак.

Вперше сутність криптоперетворення на алгебраїчних решітках у 1996 р. Запропонував M. Aitai. Решітка – це дискретна підмножина векторів (точок) у евклідовому просторі, яка є замкненою за операціями додавання та віднімання векторів. Решітка має розмірність n, якщо вона розміщується у будь-якому підпросторі простору. При геометричному поданні решітка – це множина, що рівноймовірно розміщена у просторі . За своїми властивостями алгебраїчна решітка є абелевою групою. Основою побудування алгебраїчної решітки L є множина векторів B, таких що будь-яка точка (вектор) L може бути представлений у вигляді лінійної комбінації елементів B. Стійкість LB-кріптосистем базується на складності вирішення двох комбінаторних проблем:

– найкоротшого вектору, вирішення якої зводиться до пошуку найкоротшого вектору у решітці L з базисом B;

– найближчих векторів, коли для заданого базису B решітки L и деякого вектору необхідно знайти вектор , що є найближчим до вектору v.

Вказані проблеми у загальному випадку є експоненційно складними, наприклад для більшості базисів B. Якщо базисні вектори є короткими та ортогональними, то обидві проблеми можна ефективно вирішити. Пошук таких базисів на решітці називається редукцією базису решітки і для цього можна використати відомий алгоритм LLL.

Вигідним в задачі з решітками є те, що всі ключі однаково важко зламати як в найпростішому випадку, так і в найгіршому випадку при налаштуванні будь-яких параметрів криптосистеми на основі решітки. У криптосистемі, подібній RSA, генерування ключів включає в себе вибір двох дуже великих випадкових чисел, які

повинні бути простими і повинні відповідати жорстким вимогам проблеми факторизації, але існує певний рівень ймовірності неправильного вибору і, як наслідок, слабкого рівня безпеки. У криптографії на основі решіток всі можливі ключові вибори сильні і складні до розв’язання.

LB-криптосистеми можна поділити на дві такі групи:

– імовірно нестійкі, але ефективні криптосистеми, до яких можна віднести LB-криптосистеми, що грунтуються на перетвореннях в фактор-кільцях, наприклад NTRU-алгоритм;

– імовірно стійкі, але не ефективні LB-криптосистеми. Як приклад такої криптосистеми можна назвати LWE-проблему (Learning with Errors). Складність такої криптосистеми (Ring-LWE) грунтується на комбінаторній проблемі, яка дозволяє будувати криптосистеми, що є відносно ефективні і стійкі до атак із застосуванням квантових комп’ютерів.

Основною проблемою серед усіх задач теоріх решіток (SVP) є пошук найкоротшого ненульового вектора всередині решітки. Ця проблема є NP-складною. І на відміну від проблеми факторизації чи задачі дискретного логарифмування, не існує відомого квантового алгоритму для вирішення SVP за допомогою квантового комп'ютера. На практиці криптосистеми базуються на припущенні, що полегшені варіанти цієї проблеми все ще важко вирішити. Серед усіх кандидатів NTRU та LWE показують найкращі показники ефективності та безпеки. Розглянемо кожен з них окремо.

NTRU: Криптосистема NTRU була запропонована Хофстейном як перша практична криптосистема на основі решітки. Вона ґрунтується на припущеннях, що задачі теорії решіток важко вирішити в межах певної групи решіток – решіток NTRU. У 2011 році Stehle і Steinfeld запропонували варіант схеми шифрування NTRU, SS-NTRU, який мав відношення до задач над ідеальними решітками – специфічною підгрупою решіток, що включає в себе решітки NTRU, хоч і за рахунок зниження продуктивності. Схема шифрування NTRU перевершує класичну криптографію в продуктивності, але має більші розміри публічних ключів, ніж RSA

LWE: Навчання з помилками (LWE) дозволяє криптосистемам, безпека яких зводиться до задач теорії решіток над загальними решітками. Lindner і Peikert створили криптосистему на основі решітки LP-LWE, яка є безпечною до тих пір, поки найгірші випадки проблми решіток є складними. На практиці, як правило, варіант «Навчання з помилкою» (R-LWE) використовується для підвищення ефективності. R-LWE і SS-NTRU зводяться до тієї ж задачі решіток.

У серії робіт Любашевський та ін. Запропонували схеми решітки на основі короткого цілочисельного рішення (SIS). Останнім підсумком є BLISS (схема Bimodal Lattice Signature), що є найбільш ефективною схемою підпису, що має розмір відкритого ключа приблизно 0,6 КБ і розмір приватного ключа 0,25 КБ, порівняно зі складністю AES-128. У порівнянні з RSA-2048, BLISS є приблизно в 3-10 разів швидше при генерації підпису, також BLISS швидше порівняно з RSA при перевірці. В основному BLISS є переробкою дискретно-логарифмічних підписів Schnor-a в решітки з декількома оптимізаціями по відношенню до розподілів та ефективної вибірки з цих.

2.4.2 Мультиваріативне квадратичне перетворення (MQ-криптографія)

Багатовимірні криптосистеми є системами на основі відкритих ключів і можуть використовуватися для цифрових підписів. Розглянемо сутність механізму ЕП на основі мультиваріативного квадратичного перетворення.

Відкритий ключ в мультиваріативній криптосистемі є послідовністю P1, P2,…, P2b $\in$ F2 [ω1, …, ω4b]. Із 2b поліномів з 4b змінних. ω1, …, ω4b з коефіцієнтами у полі F2 $\in$ {0,1}. Кожний поліном має мати що найбільш ступень 2, без квадратичних термів, та представлений як послідовність 1, ω1, …, ω4b, ω1ω2, ω1ω3, …, ω4b-1ω4b.

В цілому, відкритий ключ має 16b3+4b2+2b бітів. Наприклад, для b = 128, об’єм ключа складатиме 4Мегабайт. Суттєвою перевагою ЕП на основі MQ-

криптографії, наприклад над HB-підписом, є те, що підпис є коротким. Інші MQ-системи мають ще коротші підписи і у ряді випадків більш короткий відкритий ключ. Основною проблемою для криптоаналітика відносно MQ-криптографії є пошук послідовності з 4b ω1, …, ω4b бітів, що породжує 2b вище зазначених вихідних бітів (P1(ω1, …, ω4b), …, P2b(ω1, …, ω4b)).Ймовірність вгадування послідовності з 4b бітів можна оцінити як 2^{-2b}. Зафіксуємо стандартний незвідний поліном φ ∈ F2(t) ступеня 3b.

Визначимо L як поле $F2(t)/\varphi$ розмірністю 2^{3b}. Критичним кроком під час формування підпису є пошук кореня таємного одномірного (uniraviate) поліному малого ступеня над L, а саме, полінома в L[x] зі ступенем не більше ніж 2b. Існує декілька стандартних алгоритмів для вирішення цієї задачі за час $b^{O(1)}$. Таємний поліном обирається таким чином, щоб мати всі ненульові експоненти виду 2^i+2^j або 2^i. Якщо елементи x ∈ L представлені у вигляді x0+x1t1+…+x3b-1, де x_i ∈ F2, тоді $x_2=x_0+x_1t_2+\ldots+x_{3b-1}t_{6b-2}$, $x_4= x_0+x_1t_4+\ldots+x_{3b-1}t_{12b-4}$ і так далі.

Таким чином, $x_{2i}+2_j$ є квадратичним поліномом із змінними x_0, …, x_{3b-1}. Деякі прості перетворення приховують структуру цього полінома і породжують відкритий ключ. Таємний ключ підписувача має три компоненти:

– оборотну матрицю S розмірністю 4b * 4b з коефіцієнтами в F_2

– поліном Q ∈ L [x, $\nu_1, \nu_2, \ldots, \nu_b$], де кожний терм має одну з шести можливих форм, що зазначені на формулах 2.1, 2.2 та 2.3;

$$lx^{2^i+2^j}, l \in L, 2^i < 2^j, 2^i + 2^j \leq 2b \tag{2.1}$$

$$lx^{2^i}\nu_j, l \in L, 2^i \leq 2b \tag{2.2}$$

$$\begin{gathered} l\nu_I\nu_j \\ lx^{2^i} \\ l\nu_j \\ l \end{gathered} \tag{2.3}$$

– матрицю T розмірністю 2b * 3b рангу 2b з коефіцієнтами у F_2.

Підписувач обчислює відкритий ключ таким чином: $(x_0, x_1, \dots, x_{3b-1}, \nu_1, \nu_2, \dots, \nu_b)$ як S раз вектору $(\omega_1, \dots, \omega_{4b})$ всередині фактор-кільця $L[\omega_1, \dots, \omega_{4b}]/(\omega_1{}^2 - \omega_1, \dots, \omega_{1b}{}^2 - \omega_{4b})$. Обчислює $x = \sum x_b t^i$ та $y = Q(x, \nu_1, \nu_2, \dots, \nu_b)$. Подає у у вигляді $Y_0 + Y_1 t + \cdots + Y_{3b-1} t^{3b-1}$, де кожний $Y_i \in F_2[\omega_1, \dots, \omega_{4b}]$, та обчислює $(P_1, P_2, \dots, P_{2b})$. Підпис є зворотнім. Він здійснюється наступним чином:

– починаючи з величини $P_1, P_2, \dots, P_{2b}$, вирішуємо таємне лінійне рівняння $T(Y_0, Y_1, \dots, YP_{3b-1}) = (P_1, P_2, \dots, P_{2b})$ для того, щоб отримати значення $(Y_0, Y_1, \dots, YP_{3b-1})$. Існує 2^b можливих варіантів рішення для $(Y_0, Y_1, \dots, YP_{3b-1})$. Обираємо випадковим чином одне із них.

– обирається випадково значення $\nu_1, \nu_2, \dots, \nu_b, \in F_2$ і підставляємо його у таємний поліном $Q(x, \nu_1, \nu_2, \dots, \nu_b)$, отримуючи поліном $Q(x) \in L[x]$.

– обчислюється $Y = Y_0 + Y_1 t + \cdots + Y_{3b-1} t^{3b-1} \in L$ та вирішується $Q(x) = Y$, отримуємо $x \in L$. Якщо існує декілька коренів, починаємо процес з початку.

– записуємо x як $x_0 + x_1 t + \cdots + x_{3b-1} t^{3b-1}$, де $x_0, x_1, \dots, x_{3b-1} \in F_2$. Вирішуємо таємне рівняння $S(\omega_1, \dots, \omega_{4b}) = (x_0, x_1, \dots, x_{3b-1}, \nu_1, \nu_2, \dots, \nu_b)$ та отримуємо підпис.

Далі, HFE-приховане рівняння в полі $Q(x) = Y$ з пропуском декількох бітів. Тобто $Q(x) = Y$ з еквівалентом 3b рівнянь, але публікується тільки 2b рівнянь, де ν – означає «vinegar», змінні $\nu_1, \nu_2, \dots, \nu_b$. Відмітимо, що HFE перетворення, тобто без пропусків бітів та без ν-змінної може бути атаковано за $2^{(\log_{10} b)^2}$ операцій при застосуванні атаки Grobner, але HFE^{ν}-перетворення буде протистояти такій атаці.

2.2.3 Електронний підпис на основі хеш-функцій (HB-криптографія)

Механізми ЕП на основі хеш-функцій грунтуються на використанні стандартної криптографічної хеш-функції H, наприклад можна формувати хеш-

значення з довжиною від 28 до b=128 бітів. Якщо для цього обрати хеш-функцію SHA-256, то відкритий ключ підписувача в ній буде мати $8b^2$ бітів. Для вказаного значення b=128 відкритий ключ буде мати довжину у 16 кілобайт та складатиметься з 4b строчок $Y_1[0], Y_1[1], Y_2[0], Y_2[1], \dots, Y_{2b}[0], Y_{2b}$, кожна з яких має довжину 2b бітів. За цих умов, тобто для b=128, електронний підпис повідомлення m матиме довжину $2b(2b+1)$ бітів, тобто 8 кілобайт. Підпис складається зі 2b-бітових строчок $r, x_1, \dots, x_b$, таких, що біти $(h_1, \dots, h_{2b})$ хеш-значення $H(r, m)$ визначається як $Y_1[\,h_1] = H(\,x_1), Y_2[h_2] = H(\,x_2) \dots\, Y_{2b}[h_{2b}] = H(\,x_{2b})$.

З початку підписувач генерує секретний параметр x, а потім обчислює $Y = H(x)$. Особистий ключ підписувача має $8b^2$ бітів, а саме - $4b$ незалежних однорідних випадкових строк $x_1[0], x_1[1], x_2[0], x_2[1], \dots, x_{2b}[0], x_{2b}[1]$, кожна з яких містить $2b$ бітів.

Після підписувач обчислює відкритий ключ $Y_1[0]$, $Y_1[1]$, $Y_2[0]$, $Y_2[1], \dots, Y_{2b}[0], Y_{2b}[1]$ у вигляді $H(x_1[0])$, $H(x_1[1])$, $H(x_2[0])$, $H(x_2[1])$, $\dots, H(x_{2b}[0])$, $H(x_{2b}[1])$. При підписуванні повідомлення m підписувач генерує однорідну випадкову строчку T, обчислюючи біти $h_1, \dots, h_{2b}$ хеш-значення $H(r, m)$ та формує значення $(r, x_1[h_1], \dots, x_{2b}[h_{2b}])$, що і є електронним підписом повідомлення m. Далі підписувач знищує значення x, а значить більше не використовує його. Тобто це механізм, схожий на механізм одноразового електронного підпису Lomport-Diffie.

Якщо необхідно підписати більше, ніж одне повідомлення, то можна використовувати «зчеплення». В цьому випадку для підписання наступного повідомлення підписувач знову використовує згенерований відкритий ключ.

При верифікації перевірник перевіряє перше підписане повідомлення, а також включає до нього новий відкритий ключ. Після цього він зможе здійснити перевірку підпису наступного повідомлення. Підпис n-го повідомлення містить всі n-1 попередньо підписаних повідомлень. Криптографія на основі хеш-функцій досить перспективний апарат, що може бути використаний у постквантових системах цифрового підпису. Криптографія на основі хеш-функцій може

використовувати будь-яку важко обернену функцію в безпечну систему підпису на відкритих ключах. Це дозволить ефективно протистояти атаці, що грунтується на основі використання алгоритму Гровера.

2.2.4 Криптографічне пертворення з використанням збиткових кодів (СВ-криптографія)

Вважатимемо, що ціле b є ступенем двійки. Запишемо параметри збиткового коду $N = 4b\log_{10} b; d = [\log_{10} n]; t = 149$. Відкртим ключем в такій системі є матриця K розмірністю $dt * n$ з коефіцієнтами у полі F_2. Повідомлення представляє собою n-бітну строчку з вагою t, тобто n-бітна строка у своєму складі має точно t бітів, які дорівнюють 1. Для цього необхідно здійснити попереднє форматування повідомлення. Далі при зашифруванні повідомлення m відправник перемножує матрицю K на m та формує таким чином dt-бітний шифротекст $C - Km$. Головною проблемою для зловмисника при криптоаналізі є пошук синдрому декодування матриці K. Тобто, для того щоб розв'язати цю систему рівнянь, необхідно знайти вхідний вектор з вагою t. Це можна зробити методами лінійної алгебри. Для цього необхідно виконати зворотню операцію відновлення Km для деякого n-бітного вектору ν, такого що $K\nu = Km$. Але для таких векторів існує експоненційно велика кількість векторів ν, тому пошук серед них t-вектора є також експоненційно складним. Найменш складною відомою атакою є атака зі складністю e^b для більшості матриць K.

При верифікації отримувач генерує відкритий ключ K з таємною структурою, а саме – з структурою прихованого коду Гоппа. Це дозволить отримувачу провести декодування у прийнятний час. Зрозуміло, що порушник може виявити структуру прихованого коду Гоппа у відкритому ключі, але на цей час така атака невідома.

Більш конкретно. Для випадку шифрування отримувач починає з особливих елементів $\lambda_1, \lambda_2, \dots, \lambda_n$ у полі $F_2 d$ та таємного незвідного полінома системи t у полі $g \in F_2 d[x]$. Основною складовою отримувача є формування синдрома декодування, тобто матриці розмірністю $dt \times n$, що показана в формулі 2.4.

$$H = \begin{pmatrix} \frac{1}{g(\lambda_1)} & \dots & \frac{1}{g(\lambda_n)} \\ \dots & \ddots & \dots \\ \frac{{\lambda_1}^{t-1}}{g(\lambda_1)} & \dots & \frac{{\lambda_1}^{t-1}}{g(\lambda_n)} \end{pmatrix} \tag{2.4}$$

де H – це матриця синдрому декодування;

g – поле незвідного полінома;

λ – елементии поля змінних.

В такій матриці кожен елемент є елементом поля $F_2 d$ та розглядається як стовбчик із d елементів поля F_2 у стандартному базисі $F_2 d$. Ця матриця є частково контрольною матрицею для незвідного двійкового коду Гоппа і може бути декодована за допомогою алгоритму Патерсона або іншого швидкого алгоритму. Відкритий ключ K є версією матриці H.

Таємний (особистий) ключ також містить матрицю S, що інвертується, розмірністю $dt \times dt$ і матрицю перестановки P розмірністю $n \times m$. Відкритий ключ є результатом спеціального перетворення. Для того, щоб отримати HPm, отримувач перемножує шифротекст $Km = SHPm$ на S, а потім декодує H, щоб отримати Pm. Далі він перемножує результат на матрицю P^{-1} та отримує відкритий текст m.

Особливості реалізації еклктронного підпису розглядаємо на прикладі криптосистеми Нидерайтера, що має назву CFS ЕП. Доведення стійкості такої схеми зводиться до оцінки складності розвязку задачі синдромного декодування. Так, при відомому таємному ключі при декодуванні можна вирішити тільки для деякої долі випадкових слів. Для цього використовується багаторазове гешування повідомлення, яке рандомізоване засобом цього гешування з використанням лічильника з довжиною в r бітів. При цьому підписувач повинен використати свій

ключ для визначення відповідного вектора помилки. Далі таємний ключ використовується для визначення відповідного вектора помилок. Наостанок підписувач використовує цей вектор разом із значенням лічильника в якості підпису. Розглянемо сутність та дамо оцінку стійкості такого механізму ЕП.

Спочатку генеруються загальні системні параметри: $m, t \in N$. Потім генерується ключ генерації пари ключів як в системі Нідерайтера. Для цього використовується алгебраїчний код на основі (n = 2m, k = n − mt, 2t + 1) двійкових незвідних поліномів Гоппи. Далі для нього породжуються наступні матриці:

– матриця H: $(n-k) \times n$, що є матрицею перевірки алгебраїчного коду, що має t виправляючу спроможність;

– матриця X: $(n-k) \times (n-k)$ – випадкова обернена матриця;

– матриця P: $n \times m$ – випадкова матриця перестановок;

– матриця $H_X = X * H * P$ – відкритий ключ та t – виправляюча спроможність.

– матриці X та P є таємним ключем.

Далі йде обчислення підпису. Ввідними даними для обчислення підпису є наступні:

– h – функція гешування, застосовується гешування вхідних даних x, результатом є хеш-значення $h(x)$ з довжиною $n - k$ біт;

– поліноміально-складний алгоритм декодування алгебраїчного коду, що застосовується до синдромної послідовності $s = (s_0, s_1, \dots, s_{n-k-1})$;

– відкритий текст M, для якого потрібно обчислити електронний підпис.

Вихідними даними є безпосередньо сам електронний підпис Y згідно механізму CFS для відкритого повідомлення M. Сутність алгоритму електронного підпису розриємо нижче:

– обчислення хеш-значення $h(\mathrm{M})$ та присвоєння змінній i значення $i = 1$;

– обчислення хеш-значення $h(h(\mathrm{M}))$ та його конкатенація;

– значення $h(h(\mathrm{M}))$ розглядається як синдром у вигляді послідовності $s_x = (s_0, s_1, \dots, s_{n-k-1})$, що обчислена для деякого довільного кодового слова та вектора помилок $e = (e_0, e_1, \dots, e_{n-1})$;

– обчислюється значення вектору $\dot{S}_X{}^T = X^{-1} \cdot S_X{}^T$, та вектору $\bar{e}^T = P \cdot e^T$;

– для послідовності (синдрому) $\dot{S}_X$ здійснюється швидке декодування. Реалізується виконання швидкого алгоритму декодування та у разі успіху змінній присвоюється значення $i = i + 1$ та здійснюється перехід до другого пункту алгоритму;

– обчислюється значення вектору $e^T = P^{-1} \cdot e^T = P^{-1} \cdot P \cdot e^T$;

– обчислюється електронний підпис $Y = (e, i)$ для повідомлення M згідно зі схемою CFS;

– формування електронного цифрового підпису за схемою CFS $Y = (e, i)$ для відкритого текста M.

Далі розглянемо алгоритм перевірки (верифікації). Вихідними даними для перевірки є наступні:

– відкритий ключ у вигляді матриці $H_X = X * H * P$ та число t – виправляюча спроможність.

– відповідна функція гешування h;

– швидкий з поліноміальною складністю алгоритм декодування;

– значення електронного підпису $Y = (e, i)$;

– відкритий текст M.

Алгоритм верифікації зводиться до виконання наступної послідовності кроків:

– обчислення вектору здійснюється згідно механізму CFS $s'_X = H_X \cdot e^T$;

– обчислюється значення $s''_X = h(h(M) \| i)$;

– приймається рішення про правильність чи помилковість у вигляді: якщо $s'_X = s''_X$, то підпис правильний, якщо ж ні – помилковий.

Стійкості механізму відповідно до сучасних поглядів вважається експоненційно складною.

2.2.5 Криптографія на основі ізогеній еліптичних кривих

Ізогенія – це раціональне відображення $\varphi: E_1(K) \to E_2(K)$, де $E_1(K)$ та $E_2(K)$ є еліптичними кривими, а $\varphi(P_\infty) = P_\infty$[32 – 33]. Нульова ізогенія – це ізогенія, що відображає усі точки однієї кривої, у точку на нескінченності іншої. Ядром ізогенії є $Ker(\varphi) = \{K_i \in E_i\};\ \varphi(K_i) = P_\infty$.

Ізогенії ініціюють відображення полів функцій на кривих. Степінь розширення $K(E_1): \varphi * K(E_2))$ називають степінню ізогенії.

Для ізогенії $\varphi: E_1(K) \to E_2(K)$ існує дуальна ізогенія $\hat{\varphi}: E_2(K) \to E_1(K)$, така, що $\hat{\varphi}\varphi \coloneqq [l]$, де l – множення точки кривої E_1 на число l, аналогічно $\varphi\hat{\varphi} \coloneqq [l]$, де l – множення точки кривої E_2 на число l. Дуальні ізогенії мають однакову степінь.

У випадку алгебраїчно замкненого поля операція множення точки на число l задає ендоморфізм еліптичної кривої з ядром l^2 точок. Оскільки ізогенія відповідає квадратному кореню з операції множення на l, то ядро ізогенії складається з l точок порядку l, що створюють циклічну групу (однією з них є точка P_∞).

Ізогенії складних степенів можуть використовуватися як композиція ізогеній простих степеней. Властивості ізогеній, що икористовуються при створенні криптосистем:

– $\gamma\big(\varphi(A)\big) = \varphi\big(\gamma(A)\big) = \gamma\varphi(A)$;

– $k * A_\varphi = \varphi(k * A)$.

Знаходження ізогеній по ядру. Для знаходження ізогенії $\varphi: E_1(K) \to E_2(K)$ з заданим ядром викорстовується вормула Велу [14]. Для еліптичної кривої, що задана формулою Вейєрштрасса: $E_1: y^2 + a_1xy + a_3y = x^3 + a_1x^2 + a_4x + a_6$. Нехай C – группа точок еліптичної кривої, що буде ядром ізогенії. Тоді:

а) Формуємо набір точок S:

1) Виключаємо з C точку на нескінченності.

2) Нехай C_2– усі точки C, в яких координата у дорівнює нулю, а R – усі інші точки C.

3) Розділимо точки набору R на R_+ та R_- таким чином, що для кожної точки P, що належить R_+ , $-P$ належить R_-.

4) $S = R_+ \cup C_2$.

б) Для кожної точки $Q \in S$ виконуємо наступні обчислення:

1) $g_Q^x = 3x_Q^2 + 2a_2x_Q + a_4 - a_1y_Q$;

2) $g_Q^y = -2y_Q - a_1x_Q - a_3$;

3) $v_Q = \begin{cases} g_Q^x, 2Q = \infty \\ 2g_Q^x - a_1g_Q^y, 2Q \neq \infty \end{cases}$;

4) $u_Q = (g_Q^y)^2$;

5) $v = \sum_{Q \in S} v_Q$;

6) $w = \sum_{Q \in S}(u_Q + x_Q v_Q)$.

в) Розраховуємо формулу еліптичної кривої $E_2(K)$:

1) $A_1 = a_1$;

2) $A_2 = a_2$;

3) $A_3 = a_3$;

4) $A_4 = a_4 - 5v$;

5) $A_6 = a_6 - (a_1^2 + 4a_2)v - 7w$;

6) $E_2: y^2 + A_1xy + A_3y = x^2 + A_2x^2 + A_4x + A_6$.

г) Розраховуємо координати точки $(x_\varphi, y_\varphi) = \varphi(x, y)$ за формулами 5 та 6.

$$x_\varphi = x + \sum_{Q \in S}\left(\frac{v_Q}{x - x_Q} + \frac{u_Q}{(x - x_Q)^2}\right) \quad (2.5)$$

$$y_\varphi = y - \sum_{Q \in S}\left(u_Q \frac{2y + a_1x + a_0}{(x - x_Q)^3} + \frac{a_1(x - x_Q) + y - y_Q}{(x - x_Q)^2} + \frac{a_1u_Q - g_Q^x g_Q^y}{(x - x_Q)^2}\right) \quad (2.6)$$

Отримані на останньому кроці формули 5 та 6 будуть ізогенією з заданим ядром. Оскільки в криптографії еліптичних кривих прийнято використовувати

спрощену формулу Вейєрштрасса, то приведемо до прийнятого вигляду - $E_1: y^2 - x^3 + a_4x + a_6$.

Нехай C – группа точок еліптичної кривої, що буде ядром ізогенії. Тоді Формуємо набір точок S. Виключаємо с Cточку на нескінченності. Нехай C_2 – усі точки C, в яких координата y дорівнює нулю, а R – усі інші точки C.

Розділимо точки набору R на R_+ та R_- таким чином, що для кожної точки P, що належить R_+ , $-P$ належить R_-. $S = R_+ \cup C_2$. відповідно Для кожної точки $Q \in S$ виконуємо обчислення за формулами 2.7, 2.8 та 2.9. У даних формулах послідовно обчислюються значення, що отримуються у попередніх к формулах, тобто результат формули 2.7 буде використано у формулі 2.8, а результат обчислень за формулою 2.8 буде використано при розрахунку кінцевого результату за формулою 2.9.

$$\begin{aligned} g_Q^x &= 3x_Q^2 + a_4, \\ g_Q^y &= -2y_Q \end{aligned} \tag{2.7}$$

У формулі 2.8 вискористовуються значення g_Q^x та g_Q^y, що були отримані у вще наведеній формулі 2.7. У формулі 2.9 використовуємо v_Q та u_Q, що отримали у формулі 2.8.

$$\begin{aligned} v_Q &= \begin{cases} g_Q^x, 2Q = \infty \\ 2g_Q^x, 2Q \neq \infty \end{cases}, \\ u_Q &= (g_Q^y)^2 \end{aligned} \tag{2.8}$$

$$\begin{aligned} v &= \sum\nolimits_{Q \in S} v_Q, \\ w &= \sum\nolimits_{Q \in S} (u_Q + x_Q v_Q) \end{aligned} \tag{2.9}$$

Далі розраховуємо формулу еліптичної кривої $E_2(K)$ за формулою 2.10, використовуючи значення w та v, що були отримані в результаті обчислення формули 2.9.

$$\begin{gathered} A_4 = a_4 - 5v, \\ A_6 = a_6 - 7w, \\ E_2 : y^2 = x^2 + A_4 x + A_6 \end{gathered} \tag{2.10}$$

Розраховуємо координати точки $(x_\varphi, y_\varphi) = \varphi(x, y)$ за фомулою 11.

$$x_\varphi = x + \sum_{Q \in S} \left(\frac{v_Q}{x - x_Q} + \frac{u_Q}{(x - x_Q)^2}\right), \tag{2.11}$$

$$y_\varphi = y - \sum_{Q \in S} \left(u_Q \frac{2y}{(x - x_Q)^3} + v_Q \frac{y - y_Q}{(x - x_Q)^2} - \frac{g_Q^x g_Q^y}{(x - x_Q)^2}\right)$$

3 ХАРАКТЕРИСТИКА ПОСТКВАНТОВИХ АЛГОРИТМІВ ЦИФРОВОГО ПІДПИСУ

Як було згадано раніше, у серпні 2016 року NIST США ініціював початок стандартизації постквантових алгоритмів й дав один рік часу на подання своїх алгоритмів кандидатами. У листопаді 2017 року на розгляд NIST було представлено 82 алгоритми-кандидата. Серед них 69 відповідали як мінімальним критеріям прийняття, так і вимогам до подання і були прийняті в якості кандидатів першого туру в 20-го грудня 2017 року, знаменуючи початок першого раунду процесу стандартизації постквантової криптографії NIST. З 69 кандидатів до другого туру пройшли 26 алгоритмів, про що було оголошено 30 січня 2019 року [15]. Серед цих кандидатів було 18 алгоритмів-кандидатів на звання постквантового електронного підпису. До першого раунду подавались та пройшли наступні алгоритмі для цифрового підпису: Picnic, WalnutDSA, Rainbow, MQDSS, LUOV, HiMQ-3, Gui, GeMSS, DualModeMS, SPHINCS+, Gravity-SPHINCS, pqsignRM, RaCoSS, qTesla, pqNTRUSign, FALCON, DRS, CRYSTAL-DILITHIUM.

Було визначено три широких аспекти критеріїв оцінки, які використовувались для порівняння алгоритмів-кандидатів напротязі всього процесу стандартизації NIST PQC. Ці аспективключають наступн пункти: безпека, вартість і продуктивність, алгоритм і характеристики реалізації. Дані вимоги були описані вище у першому розділі.

NIST відібрав 26 кандидатів другого туру з 69 кандидатів першого туру, використовуючи критерії оцінки, що були зазначені вище. Для оцінки безпеки алгоритму NIST розглянув аргументи безпеки, представлені в пакеті, що був наданий кандидатом, а також зовнішній криптоаналіз, представлений NIST або опублікований в іншому місці. Дослідники NIST також провели внутрішній криптоаналіз.

NIST розглядав не тільки атаки, які прямо демонстрували, що кандидат не дотягує до заявлених цілей безпеки NIST, але й атаки, які ставили під сумнів

основоположні принципи безпеки кандидата, або ж ті атаки, які знаходили пункти, які можна покращити в алгоритмі. NIST також розглянув загальну кількість, якість і зрілість аналізу для кожного кандидата, включаючи аналіз аналогічних схем.

Після перевірки безпеки наступним найбільш важливим критерієм при відборі кандидатів другого туру була результативність. При оцінці ефективності кандидатів NIST враховував як розміри ключа/шифротекста/підпису, так і обчислювальні оцінки, що були надані кандидатами-заявниками в їх документації, що була подана разом з заявуою на участь у першій конференції по стандартизації NIST PQC. NIST також тестував продуктивность, використовуючи код з пакетів заявників.

Крім зазначеного вище, NIST розглянув зовнішні відгуки та оцінки продуктивності, надані криптографічним співтовариством. Можна відзначити, що NIST заявив, що «парамери ефективності не будуть грати важливу роль на ранній стадії процесу оцінки», і NIST не використовував показники ефективності та швидкодії реалізації як вагомого критерії пр иприйнятті свого рішення.

У кількох випадках представлений алгоритм був обраний частково за його унікальність і елегантність. NIST зазвичай віддавав перевагу тим проектам, які були засновані на чітких принципах або тим чи іншим чином демонстрували інноваційну ідею. NIST вважає, що різноманітність конструкцій дасть можливість криптографам і криптоаналітикам розширити сферу ідей у своїй галузі, а також зменшить ймовірність того, що один тип атаки усуне основну частину кандидатів, обраних в процесі стандартизації.

Алгоритми, які не були обрані для переходу до наступного раунду, не розглядаються для стандартизації NIST. Серед алгоритмів для цифрового підпису у другий раун було обрано дев’ять кандидатів що пройшли перевірку на безпеку, ефективність та вартість. Це наступні алгоритми: Picnic, Rainbow, MQDSS, LUOV, GeMSS, SPHINCS+, qTesla, FALCON, CRYSTAL-DILITHIUM [15]. Розглянемо кожен із них.

3.1 Алгоритм цифрового підпису «CRYSTAL-DILITHIUM»

Dilithium – це схема підпису на решітках, побудована з використанням евристики Фіата-Шаміра, безпека якої заснована на твердості проблеми MLWE. Dilithium входить в комплект CRYSTALS разом з механізмом обміну ключами Kyber. Основне нововведення Dilithium полягає в тому, що розмір відкритого ключа зменшується, опускаючи деякі біти низького порядку; щоб компенсувати це, кожний підпис включає в себе додаткову «підказку», яка дозволяє верифікатору перевіряти підпис. Dilithium пропонує досить хорошу продуктивність і є відносно простим в реалізації.

Найбільш відомі атаки проти Dilithium засновані на редукції решітки, без істотного використання алгебраїчної структури задачі MLWE. Вибір параметрів для Dilithium базується на консервативних оцінках вартості цих атак. Dilithium має формальний доказ безпеки в класичній моделі випадкового оракулу. Це доказ не є суттєвим і розбивається в квантовій моделі випадкового оракулу; однак поки що не відомо ніяких подібних. NIST буде заохочувати подальший аналіз всіх аспектів цієї схеми.

3.2 Алгоритм цифрового підпису «FALCON»

Falcon – це також схема підпису на решітках, заснована на GPV (Gentry-Peikert-Vaikuntanathan) Гауссовій вибірці з використанням решітки NTRU. Основне нововведення – дуже швидкий рекурсивний алгоритм гауссової вибірки, що використовує деревовидну структуру даних («Falcon-дерево»). Найбільш відомі атаки проти Falcon засновані на зменшенні решітки, без істотного використання спеціальної структури решітки NTRU. Falcon має формальний доказ безпеки в квантовій моделі випадкового оракулу. Falcon пропонує дуже хорошу

продуктивність. Однак це досить складно реалізувати, оскільки він значною мірою спирається на структуру «tower of fields» числового поля і вимагає обчислень з плаваючою точкою подвійної точності. Необхідно прикласти додаткові зусилля для тестування, щоб забезпечити, що підпис є стійким до побічних каналів атак.

3.3 Алгоритм цифрового підпису «qTESLA»

qTESLA – схема підпису на основі решітки, яка використовує припущення, що розподіл RLWE (Ring learning with errors – навчання з помилками в кільці) нічим не відрізняється від випадкових. Відкритий ключ в qTESLA, грубо кажучи, є зразком розподілу RLWE. Підписувач зберігає секретну інформацію про цей зразок розподілу і використовує цю інформацію разом з хеш-функцією для створення підписів. Перевірка підпису включає в себе деяку просту арифметику всередині обраного кільця, а потім перерахунок хеш-функції. qTESLA має досить хороші параметри продуктивності в порівнянні з іншими схемами підписів на базі решітки. Відправники qTESLA заявили про жорсткий доказ безпеки для схем у квантовій моделі випадкового оракулу. Було помічено, що помилка в доказі безпеки вимагає коригування параметрів (що знижує продуктивність схеми). Аргумент безпеки припускає гіпотезу про розподіл випадкових елементів в кільці. Враховуючи, що гіпотеза, схоже, не відповідає припущенню про безпеку, NIST сподівається краще протестувати наслідки такого припущення у другому раунді.

3.4 Алгоритм цифрового підпису «GeMSS»

GeMSS – це «big-field» багатовимірная схема цифрового підпису у добре-вивченому сімействі HFEv (Hidden Field Equations). Схема перетворює базовий

HFEv-дизайн, імовірно має екзистенціальну непростимость, в схему захищеної підпису EUF-CMA з використанням конструкції Файстеля-Патарина. Екзистенціальна вимога непростимості для HFEv-дизайну слабо пов'язана з добре вивченими MQ (багатовимірними квадратичними) і MinRank-завданнями. GeMSS пропонує деякі з найменших довжин підпису Серед всіх уявлень. GeMSS також виграє від того, що HFEv-конструкція є одним з найбільш вивчених примітивів підпису в літературі.

Крім розміру підпису та часу перевірки, деякі інші характеристики продуктивності підпису GeMSS викликають деякі побоювання. Час підпису досить великий, розмір відкритого ключа також; ці властивості можуть бути особливостями схеми GeMSS, які були успадковані від HFEv-методології. Тим не менш, аналіз безпеки показує, що можливі компроміси між ступенем прив'язки, мінус ранг проекції і кількість vinegar змінних, наприклад, які можуть вплинути на різні характеристики продуктивності. Алгоритм прийнято до другого раунду з надією на оптимізацію деяких характеристик, що значно покращить алгоритм.

3.5 Алгоритм цифрового підпису «LUOV»

LUOV – «small-field» багатовимірна схема цифрового підпису, заснована на схемі незбалансованого «Масла та оцту» (UOV - nbalanced Oil and Vinegar). Основним нововведенням схеми є визначення відкритого ключа у вигляді карти на певному кінцевому поля в той час, як коефіцієнти публікуються на суб-полі. Схема дозволяє уникнути атак, які безпосередньо використовують структуру суб-поля, використовуючи hash-and-sign підхід, гарантуючи, що розширення поля має бути використаним.

Схема також використовує псевдовипадковий генератор для побудови частини відкритого ключа, для якої може бути вирішена відповідна частина

закритого ключа, аналогічна конструкцій в циклічних UOV, циклічних радугах і їх псевдовипадкових аналогах.

LUOV пропонує явний компроміс між розміром ключа і довжиною підпису, що робить схему гнучкою для несумірних варіантів використання. Сума розміру відкритого ключа і довжини підпису є найменшою для схем багатовимірних підписів кандидатів.

Схема також має режим відновлення повідомлень. LUOV був центральним об'єктом вивчення в багатовимірній криптографії протягом двадцяти років; таким чином, електронний підпис LUOV має міцний фундамент. Однак підйомне нововведення є дуже новим, і в другому раунді безпека має бути проаналізована детальніше.

3.6 Алгоритм цифрового підпису «MQDSS»

MQDSS – це багатовимірна схема цифрового підпису, отримана з доведено безпечної схеми ідентифікації, заснованої на проблемі MQ. Схема підпису будується з ідентифікаційної схеми шляхом застосування узагальнення перетворення Фіата-Шаміра, відповідних 5-прохідних схемам ідентифікації. Відзначимо, що запропоновані параметри не задовольняють гіпотезам зниження безпеки. MQDSS підтримує генерацію псевдовипадкових ключів з великими підписами. Характеристики продуктивності MQDSS найбільше можна порівнянняти з характеристиками схем підпису на основі хешу.

Необхідно детальніше проаналізувати схему підпису MQDSS з більш агресивно підібраних наборів параметрів. Очікується, що по мірі кращого розуміння безпеки граничних випадків можлива подальша оптимізація. NIST просуває MQDSS вперед в надії, що подальші дослідження приведуть до поліпшення параметрів і продуктивності.

3.7 Алгоритм цифрового підпису «Rainbow»

Rainbow – це багатовимірна схема цифрового підпису, яка є узагальненням структури UPOV, що дозволяє параметризацію, які більш ефективна за рахунок додаткової алгебраїчної структури. Rainbow підпис в його форматі вивчається вже близько п’ятнадцяти років з різними параметрами. Подавачі Rainbow стверджує, що EUF-CMA security використовує хеш-конструкцію з «випадковою сіллю».

Спектр параметрів Rainbow дозволяє оптимізувати різні варіанти використання. Ще однією перевагою підпису Rainbow є те, що він також досліджувався в інших контекстах, в тому числі в легких додатках.

Представники Rainbow пропонують набори параметрів, націлені на всі рівні безпеки, зазначені у вимогах NIST до заявок. В якості кандидата другого раунду, очікується, що буде приділено більше уваги до більш вузького набору специфікацій.

Збільшення кількості досліджень дасть змогу зібрати методи оптимізації для Rainbow keys, які існують в літературі і які не розглядалися в уявленні Rainbow, що в ідеалі призведе покращенн схеми підпису Rainbow.

3.7 Алгоритм цифрового підпису «Picnic»

Picnic – це схема підпису, яка не використовує теоретико-числові або структуровані припущень складності. Зменшення безпеки відносяться до хеш-функцій і симетричних блокових шифрів. Підпис Picnic базується на неінтерактивному нульовому доказі знання секретного ключа. Підписується відкритий текст таким чином (через хешування), що тільки власник секретного ключа може вивести доказ і перевірити текст на валідність.

Picnic має невеликий розмір відкритого ключа, великі підписи. Підпис як і перевірка підписанного тексту досить повільні. Генерація ключів досить ефективна. Довжина підпису залежить від мультиплікативної складності схеми шифрування і від конкретної методики побудови доказу нульового знання (з області безпечних багатоваріативних обчислень).

Пікнік має доволі модульну схему проектування. Криптографічні примітиви – хеш і блоковий шифр – можуть бути створені різними способами. Представлений дизайн використовує lowmc, блоковий шифр з низькою мультиплікативною складністю.

LowMC не вивчався так багато, як AES, і, отже, потребує набагато більшого аналізу. Ефект використання AES замість lowMC в Picnic, мабуть, полягає в розширенні довжини підпису на коефіцієнт, який коливається від 6 до 9, в залежності від розміру блоку.

Поліпшення обчислень призведе до зменшення підписів. Варто зазначити, що вимоги безпеки для базового блочного шифру менш суворі, ніж загальні вимоги безпеки блокового шифру, тому що тільки одна (випадковий відкритий текст, шифротекст) пара коли-небудь розкривається.

3.9 Алгоритм цифрового підпису «SPHINCS+»

SPHINCS+ – це схема підпису на основі хешу без стану. Схеми підписів на основі хеш були вперше запропоновані в кінці 1970-х років, і з тих пір було розроблено багато поліпшень. SPHINCS+ використовує дві різні схеми підпису на основі хеш: Winternitz One-Time-Signature Plus (WOTS+), одноразову схему підпису і ліс випадкових підмножин (FORS), схему з декількома сигнатурами. Пара ключів підпису SPHINCS+ складається з 2^{60} або більше пар ключів FORS. Для перевірки аутентичності окремих відкритих ключів FORS використовується кілька рівнів хеш-дерев Merkle.

Кожен з відкритих ключів FORS і корінь кожного з дерев хеша Merkle підписується за допомогою WOTS+. Повідомлення підписуються з використанням псевдовипадково вибраного ключа FORS, а підпис повідомлення складається з підпису FORS, одного підпису WOTS+ для кожного рівня дерева Merkle і проміжних хеш-значень, необхідних для проходження кожного дерева Merkle для перевірки.

Основна перевага SPHINCS+ полягає в тому, що його безпека залежить виключно від опору зображення хеш-функції, що використовується. SPHINCS+ також має дуже маленькі відкриті ключі – від 32 до 64 байт, в залежності від рівня безпеки. Недоліком є те, що підписання повідомлення дуже повільне, а підписи відносно великі, хоча SPHINCS+ пропонує різні варіанти параметрів, які змінюють розмір підпису на швидкість. SPHINCS+ використовує псевдовипадкову функцію (PRF) для генерації ключів і бітових масок для кожного виклику хеш-функції, щоб захистити від багатоцільових атак проти хеш-функції. Більша частина вартості генерації та перевірки підпису – це час, витрачений на створення цих ключів і бітових масок. Можлива покращенням до другого раунду може бути застосування більш ефективної техніки захисту від багатоцільових атак.

Після вибору і оголошення алгоритмів, що пройшли до другого раунду, кандидатам дали час на покращення своїх алгоритмів, але тільки в тому випадку, коли ці покращення незначні і не змінюють алгоритм кардинально. Вже 15 березня 2019 року почався другий раунд процессу стандартизації постквантових алгоритмів. В найближчі 12-18 місяців представники криптографічної спільноти можуть тестувати запропоновані алгоритми на надійність, безпеку та продуктивність.

У серпні 2019 пройде конференція, де кандидати, щ запропонували дані алгоритми, можуть презентувати покращення, що були внесені до алгоритмів. В 2020 году NIST планує обрати фіналістів для останнього раунду, або обрати невелику кількість кандидатів для стандартизації.

3.10 Аналіз постквантових алгоритмів цифрового підпису й вибір алгоритму для реалізації

Автори алгоритму Dilithium провели тестування алгоритмів, що пройшли до другого раунду [16], результати наведені у таблиці 3.1. Тестування Dilithium, FALCON, qTESLA, MQDSS схем підписів провели на комп'ютері Core i7-4770K (Haswell), схеми Rainbow та GeMSS тестувалися на Intel Xeon E3-1245 v3 (Haswell), а SPINCS+ на Intel Xeon E3-1245 (Haswell). Параметр у колонці «Безпека» взятий з офіційних документацій кожного з алгоритмів.

Таблиця 3.1 – Порівняння постквантових алгоритмів

Схема підпису	Без-пека	Захищеність від timing attack (атак по часу)	Цикли підписання	Цикли перевірки підпису	Розмір публічного ключа, Bytes	Розмір підпису, Bytes
Dilithium	125	Так	789	209	1472	2701
FALCON (NTRU-GVP)	>> 128	Ні	-	-	1792	1200
qTESLA	98	Так	143402	19284	12582912	2444
GeMSS (HmFEv)	128	Так	1497	15	83100	61
LUOV	128	Так	659000	290000	34100	421
MQDSS	128	Так	8510	5752	72	40952
Rainbow	128	Так	68	22	145500	48
Picnic	128	Так	1034	194	64	195458
SPHINCS+	128	Так	51636	1451	1056	41000

Отже було обрано альтернативи, з котрих необхідно обрати один найкращий варіант, який буде реалізовано. Вибір алгоритму – це багатокритеріальна задача, тому що аналіз буде зроблено по декільком критеріям. Це наступні критерії: рівень безпеки, захищеність від timing атак, розмір публічного ключа, розмір підпису, цикли підписання повідомлення та цикли перевірки підпису. Цикли генерації

підпису не включається як критерій вибору, так як має невеликий вплив на використання алгоритму підпису. Підпис генерується лише один раз, а далі використовується протягом довгого часу, тому навіть якщо час генерації підпису є значним, то це матиме лише єдиноразовий недолік і не вплине на продуктивність роботи алгоритму в цілому.

Зхищенність від атак по часу є критичною, тому алгоритм FALCON одразу відкидається. Так як усі інші алгоритми мають однакові показники, то даний критерій відкидається, так як порівняння по ньому не дає корисної інформації. Серед критеріїв, що залишилися, усі критерії – це кількісні критерії з абсолютною шкалою [17]. Найкращий алгоритм – це алгоритм, що я найнадійнішим, найшвидшим та займає менше пам’яті, тому за критерієм безпеки буде є задача пошуку масимуму, а для інших критеріїв – пошуку мінімуму.

Виділимо множину Парето чи множину недомінантних альтернатив [18]. Для цього необхідно визначити які альтернативи гірші хоча б за одну іншу альтернативу за всіма показниками.

Проаналізувавши таблицю використовючи відношення Парето було помічено, що qTESLA є гіршою за всіма показниками у порівнянні з GeMess та Rainbow алгоритмами, тому відкидуємо цю альтернативу. Усі інші альтернативи мають різні домінантні відношення по різним критеріям, тобто складають множину недомінантних альтернатив, серед яких буде проведено подальший аналіз і вибір однієї з них. Критерії безпеки після виключення алгоритму qTESLA з альтернатив стає неважливим, так як усі альтернативи мають однакові значення за цим критерієм. В даному випадку неможливо визначити один головний критерій, так як усі параметри важливі, тому метод головного критерію для прийняття рішення не підходить.

Серед критеріїв, що будуть використовуватись для остаточної оцінки алгоритмів та вибору одного з них, є наступні: розмір публічного ключа та підпису, кількість циклів підписання та перевірки. Ці критерії кількісні й мають абсолютну шкалу вимірювання, але вони мають різні одиниці вимірювання, тому наступним кроком необхідно нормалізувати ці критерії, тобто представити значення з таблиці

так, щоб вони приймали значення від 0 до 1. Так як для критеріїв немає еталонного значення, оберемо спосіб нормування з урахуванням мінімального та максимального значень. Так як необхідно мінімізувати показники, то використаємо наступну формулу:

$$f = \frac{f_{\text{вимір}} - f_{max}}{f_{min} - f_{max}} \quad (3.1)$$

де f — це значення нормалізованого критерію;

$f_{\text{вимір}}$ – це значення, що нормалізується;

f_{max} – це найбільше з наявних значень;

f_{min} – найменше з наявних значень.

В якості приклада розглянемо нормалізацію критерія циклів підписання для алгоритму Dilithium. У даному випадку $f_{\text{вимір}} = 789$, тобто значення циклів підпису для алгоритму Dilithium, $f_{max} = 659000$ – це найбільше значення кількості циклів, й $f_{min} = 68$ тобто найменше значення серед кількості циклів підписання. Отже нормалізуємо значення циклів підписання для алгоритму Dilithium: $f = \frac{789-689000}{659000-68} = 0.99$ – це нормалізоване значення. Далі приведемо таблицю 3.2 з нормалізованими значеннями за усіма критеріями.

Таблиця 3.2 – Нормалізовані значення альтернатив алгоритмів за кожним з обраних критеріїв

Схема підпису	Цикли підписання	Цикли перевірки підпису	Розмір публічного ключа, Bytes	Розмір підпису, Bytes
Dilithium	0.9989	0.9993	0.9903	0.9864
GeMSS (HmFEv)	0.9978	1.0000	0.4291	0.9999
LUOV	0.0000	0.0000	0.7660	0.9981
MQDSS	0.9872	0.9802	0.9999	0.7907
Rainbow	1.0000	0.9999	0.0000	1.0000
Picnic	0.9985	0.9994	1.0000	0.0000
SPHINCS+	0.9217	0.9950	0.9932	0.7904

Ключовою характеристикою у роботі алгоритму є швидкість роботи, тобто найкращі (найменші) показники циклів підпису та перевірки, тому визначемо вагові коефіцієнти для критеріїв. Для циклів підписання та перевірки вагові коефіцієнти будуть рівні двум, а для розміру публічного ключа та підпису – по одиниці. Вагові коефіцієнті наведені у таблиці 3.3.

Далі виконаємо згортку критеріїв. Використаємо лінійну адитивну згортку з використанням вагових коефіцієнтів.

$$w(x) = \frac{2}{6}f_1 + \frac{2}{6}f_2 + \frac{1}{6}f_3 + \frac{1}{6}f_4 \quad (3.2)$$

де $w(x)$ – це згортка однієї альтернативи;

f_1 – це нормалізоване значення циклів підписання для альтернативи;

f_2 – це нормалізоване значення циклів перевірки;

f_3 – це нормалізоване значення розміру публічного ключа;

f_4 – це нормалізоване значення розміру підпису.

Таблиця 3.3 – Вагові коефіцієнти критеріїв

Критерій	Ваговий коефіцієнт
Цикли підписання	$\frac{2}{6}$
Цикли перевірки	$\frac{2}{6}$
Розмір публічного ключа	$\frac{1}{6}$
Розмір підпису	$\frac{1}{6}$

У таблиці 3.4 представлені результати згортки кожної з альтернатив. Найкращий показник є у алгоритму Dilithihum, що свідчить про те, що він має збалансовані високі показники за кожним з критеріїв. Наступним йде MQDSS і далі SPHINKS+. LUOV має найнижчі показники. Усі інші алгоритми мають досить

гарні показники й вище 0,8, що свідчить про те, що кожен алгоритм доволі збалансований за критеріями.

Таблиця 3.4 – Згортка критеріїв альтернатив алгоритмів

Схема підпису	Згортка критеріїв
Dilithium	0.9757
GeMSS (HmFEv)	0.8879
LUOV	0.2823
MQDSS	0.9357
Rainbow	0.8200
Picnic	0.8193
SPHINCS+	0.9179

Для реалізації алгоритму обираємо Dilithium, так як він має найкращий результат за згорткою параметрів. Жодна з критеріальних оцінок алгоритму не є найкращою і жодна з оцінок не є слабкою. Алгоритм є найзбалансованішим за швидкістю з досить гарними показниками розміру публічного ключа та підпису.

4 АЛГОРИТМ ЦИФРОВОГО ПІДПИСУ CRYSTAL-DILITHIUM ТА ЙОГО РЕАЛІЗАЦІЯ

4.1 Теоретичні відомості. Оптимізація алгоритму.

Найбільш явним, але легко-виправленим, фактором неефективності у порівнянні з іншими алгоритмами, що працюють на базі решіток, полягає в тому, що відкритий ключ складається з матриці $k \times l$ поліномів, що можуть мати значний розмір. Очевидним вирішенням даної проблеми мати матрицю A, що згенерована через заповнення поліномів p з використанням алгоритму SHAKE-128, і це є стандартним методом. Новизна алгоритму Dilithium над стандартно схемою полягає у тому, що розмір ключа зменшується у 2,5 рази за рахунок збільшення цифрового підпису на 150 байт. Для рекомендованого рівня безпеки, схема повинна мати підпис розміром 2,7KB і відкритий ключ розміром 1,5KB.

Основним спостереженням для отримання цього компромісу є те, що коли верифікатор обчислює w'_1 (коефіцієнт бітів верхнього порядку) – біти верхнього порядку, що були викорстані при формуванні підпису, не залежать від бітів низького порядку t, оскільки t перемножувалось на дуже легковісний поліном c. У схемі підпису Dilithiun деякі біти низького порядку t не входять до відкритого ключа, що призводить до того, що верифікатор не завжди може правильно обчислити біти високого порядку – $Az - ct$. Щоб компенсувати це, підписувач включає деякі «підказки» як частину підпису, що по суті є додаванням до підпису бітів низького порядку t. З цими «підказками» верифікатор зможе правильно обчислити біти верхнього порядку w'_1.

Крім того, що схему було зроблено детермінованою, використовуючи стандартну методику додавання параметра заповнення до секретного ключа і використовуючи цей параметр разом з повідомленням для запезпечення випадковості вектору поліномів y.

Останній результат Kiltz et al. [19] показав, що чим менше різних підписів бачить отримувач для тих самих повідомлень, тим сильнішим є скорочення в

квантовій випадковій моделі оракула між схемою підпису та базовими твердженнями.

Основною алгебраїчною операцією, що виконується у схемі, є множення матриці A, елементи якої є поліномами в $Zq\ [X]\ /\ (X^{256}\ +\ 1)$ вектором таких поліномів. У рекомендованому параметрі A – це матриця 5×4, що складається з 20 поліномів. Таким чином, множення Av включає в себе 20 поліноміальних множення.

Як і в більшості схем на основі решіток, які базуються на операціях над кільцями многочленів, для Dilithium було обране кільце так, щоб операція множення мала дуже ефективну реалізацію через числову теоретичну трансформацію (NTT - Number Theoretic Transform), що є різновидом DFT (дискретна трансформація Фур'є), що працює над кінцевим полем Z_q, а не над комплексними числами.

Щоб увімкнути NTT, потрібно було обрати початковий параметр q так, щоб група $Z_q\ *$ мала елемент порядку $2n\ =\ 512$; або еквівалентно $q \equiv 1\ (mod\ 512)$. 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 $a\ \in\ Z_q[X]\ /\ (X^{256}\ +\ 1)$ у своїй CRT (Chinese Remainder Theorem) сформований як $(a(r), a(r^3), \ldots, a\ (r^{2n-1}))$.

Перевагою такого представлення є те, що добуток двох поліномів є координатним. Тому найдорожчими частинами поліноміального множення є перетворення $a \rightarrow \hat{a}$ та інверсна $\hat{a} \rightarrow a$, що є NTT та інверсійними NTT-операціями.

Іншою важливою операцією, що вимагає багато часу, є розширення заповнення поліномами p поліноміальної матриці A. Матриця A необхідна як для підписання, так і для перевірки, тому ефективна реалізація SHAKE-128 є важливою для ефективності схеми.

Для AVX2 оптимізованої реалізації NTT було застосовано інший підхід, що відрізняється від інших швидких реалізацій в тому, що було використано цілочисельну арифметику. Хоча упаковується тільки 4 коефіцієнта в один

векторний реєстр з 256 біт, що є тією ж щільністю, що й використовують реалізації з плаваючою точкою, цим можна поліпшити швидкість множення приблизно в 2 рази. планування інструкцій і перемежування множень і скорочень під час NTT, так що частини затримки множення приховані.

4.2 Схема підпису

Алгоритми генерації, підписання та перевірки ключів для Dilithium схеми підпису представлені на рисунку 4.1. Під час дипломної роботи було розглянуто детерміновану версію схеми, в якій задання випадковості, що була використана в процедурі підпису, генерується (з використанням SHAKE-256) як детермінована функція від повідомлення та секретного ключа невеликого розміру. Оскільки можлива необхідність у повторенні процедури підписання, поки остаточній підпис не буде згенеровано, також додається лічильник, щоб зробити результат SHAKE-256 різним у разі підписання однакових повідомлень. Через те, що кожне повідомлення може вимагати декілька ітерацій для підпису, також з допомогою стійкої до колізій хеш-функції обчислюється початкова обробка повідомлення, що буде далі використовуватись протягом всієї процедури підпису.

Схема роботи Dilithium алгоритму базується на підході «Fiat-Shamir with Aborts» [20] і найбільше схожий на схеми, запропоновані в [21], що є не дуже ефективним. Классичний алгоритм генерації ключів генерує матрицю A розміром $k \times l$, у якій кожен елемент це поліном у кільці $R_q = \frac{Z_q[X]}{X^n+1}$. Для всіх рівнів безпеки буде використовується кільце з параметрами $q = 2^{23} - 2^{13} + 1$ й $n = 256$. В різних рівнях безпеки використовується більше/менше операції над кільцем та більше/ менше операцій XOF (extendable-output function – хеш-функція змінної довжини, оj застосовується на повідомленні, в якому вихід може бути розширений до будь-якої бажаної довжини). Після цього алгоритм відбирає випадкові секретні

ключі векторів s_1 і s_2. Кожен коефіцієнт цих векторів є елементом кільця R_q з малими коефіцієнтами – розміром не більше η (статичне число, що задається в залежності від обраного рівня безпеки). Друга частина відкритого ключа обчислюється як $t = As_1 + s_2$. Всі алгебраїчні операції в цій схемі вважаються такими, що можуть бути застосовані над поліноміальним кільцем R_q.

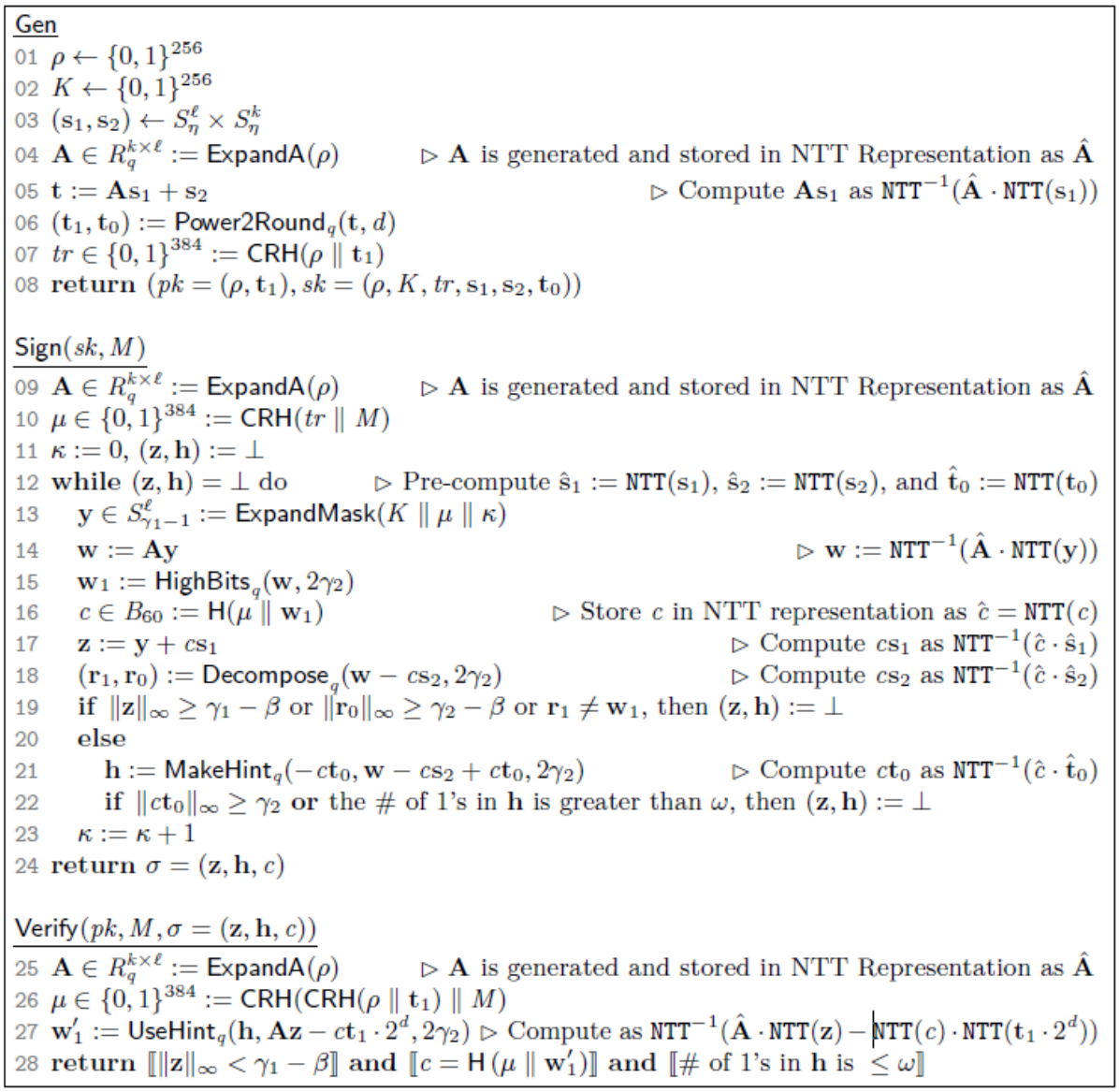

Рисунок 4.1 – Схема підпису Dilithium

Основне конструктивне удосконалення алгоритму Dilithium у порівнянні з класичною схемою, полягає в тому, що розмір відкритого ключа зменшується приблизно в 2,5 рази за рахунок додання додаткових 100 байт до підпису. Для

зменшення розміру алгоритм генерації ключа виводить $t_1 := Power2Round_q(t, d)$ в якості відкритого ключа замість t, як це робиться в классичній схемі. Це означає, що замість $[log\ q]$ біт на коефіцієнт, публічний ключ вимагає $[log\ q] - d$ біт. У алгоритмі Dilithium $q \approx 2^{23}$ і $d = 14$, що означає, що замість 23 біт у кожному коефіцієнті відкритого ключа використовується 9 біт.

Щоб зберегти розмір відкритого (і секретного) ключа невеликим, процедури підписання та верифікації починаються з заповнення поліномами матриці A (або, точніше, її NTT представлення у вигляді – $\hat{A}$). Якщо місце зберігання матриці не є принциповим, то можна заздалегідь обчислити $\hat{A}$ й використовувати її як частину секретного/відкритого ключа. Користувач може додатково попередньо обчислити s_1, s_2, t_0 у представленні NTT, щоб трохи прискорити операцію підпису. З іншого боку, якщо користувач хоче зберегти якомога менше інформації про секретний ключ, йому потрібно тільки зберегти кільце поліномів p і K, а також випадкове занчення seed, що використовується для створення s_1, s_2 в алгоритмі генерації ключів. Всі інші частини секретного ключа можуть бути відтворені з них. Крім того, можна також зменшити необхідний об'єм пам'ять при проміжних обчисленнях, лише зберігаючи частині NTT представлення, з яким в даний час працює алгоритм.

Ще одна можлива зміна полягає в усуненні жорсткої детермінованої природи цифрового підпису. Можливо, варто розглянути цей варіант через side-channel атаки, що використовують детермінізм [22]. Простий спосіб реалізувати можливість використання рандомізованих підписів полягає у тому, щоб дозволити додавання деякої системної випадковості на вхід $ExpandMask$ при генерації y. Доказ безпеки алгоритму Dilithium жорсткий відповідно до [19] для детермінованих сигнатур, і рівень стійкості поступово слабшає, відповідно до збільшення різних підписів на повідомлення. Тому рекомендовано використовувати детерміновані сигнатури, за винятком середовищ, які можуть бути уразливі для вищезазначених атак бічних каналів.

4.3 Реалізація алгоритму

В якості мови для реалізації алгоритму було обрано мову програмування Golang [23]. Це відносно нова мова програмування, але вже досить поширена й з кожнім роком стає більш популярною. Golang є простим у вивченні та використанні, процес розробки є дуже простим й не вимагає платного средовища розробки. Програми написані на Golang мають гарні показники за продуктивністю й швидкістю виконная. Також на Golang є бібліотеки, що реалізовую необхідні криптоалгоритми й математичні операції. Golang є кроссплатформенним, а також є можливість скомпілювати пакети на Go у C бібліотеки, що дозволить зінтегрувати код з такими мовами як C, Python, Java, Ruby, Node. Через перспективність мові програмування, простоту й зручність у використанній, гарні показники роботи додатків на цій мові, можливість інтеграції в усі платформи було обрано Go для реалізації алгоритму Dilithium. Бібліотека дозволяє виконати сценарій, що відображений на рисунку 4.2.

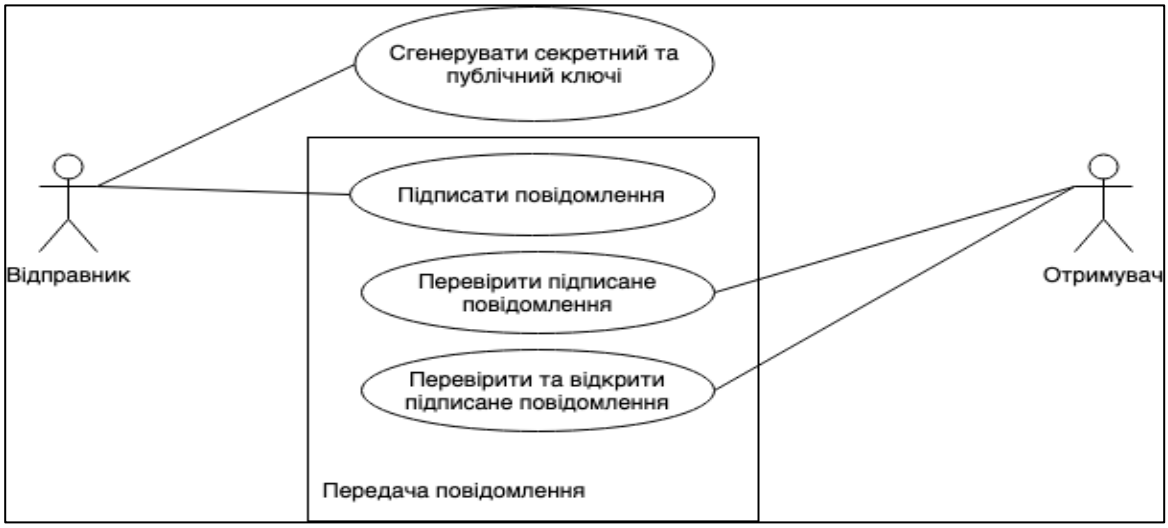

Рисунок 4.2 – Use case діаграма додатку

Припустимо, що бібліотека буде вікористана безпосередньо для генерації ключів та передачі повідомлення. Тоді Відправник може сгенерувати секретний та публічний ключі, котрі будуть використані для підпису та перевірки повідомлення.

Відправник може підписати повідомлення за допомогою отриманого секретного ключа. А отримувач, якому був предоставлений публічний ключ, може перевірити поведомлення на цілісність чи одразу відкрити повідомлення, якщо підпис був валідний. Сторонній код, що використовуватиме бібліотеку, може розширити даний сценарій. Далі розглянемо діаграму класів реалізованого алгоритму, що зображена на рисунку 4.3.

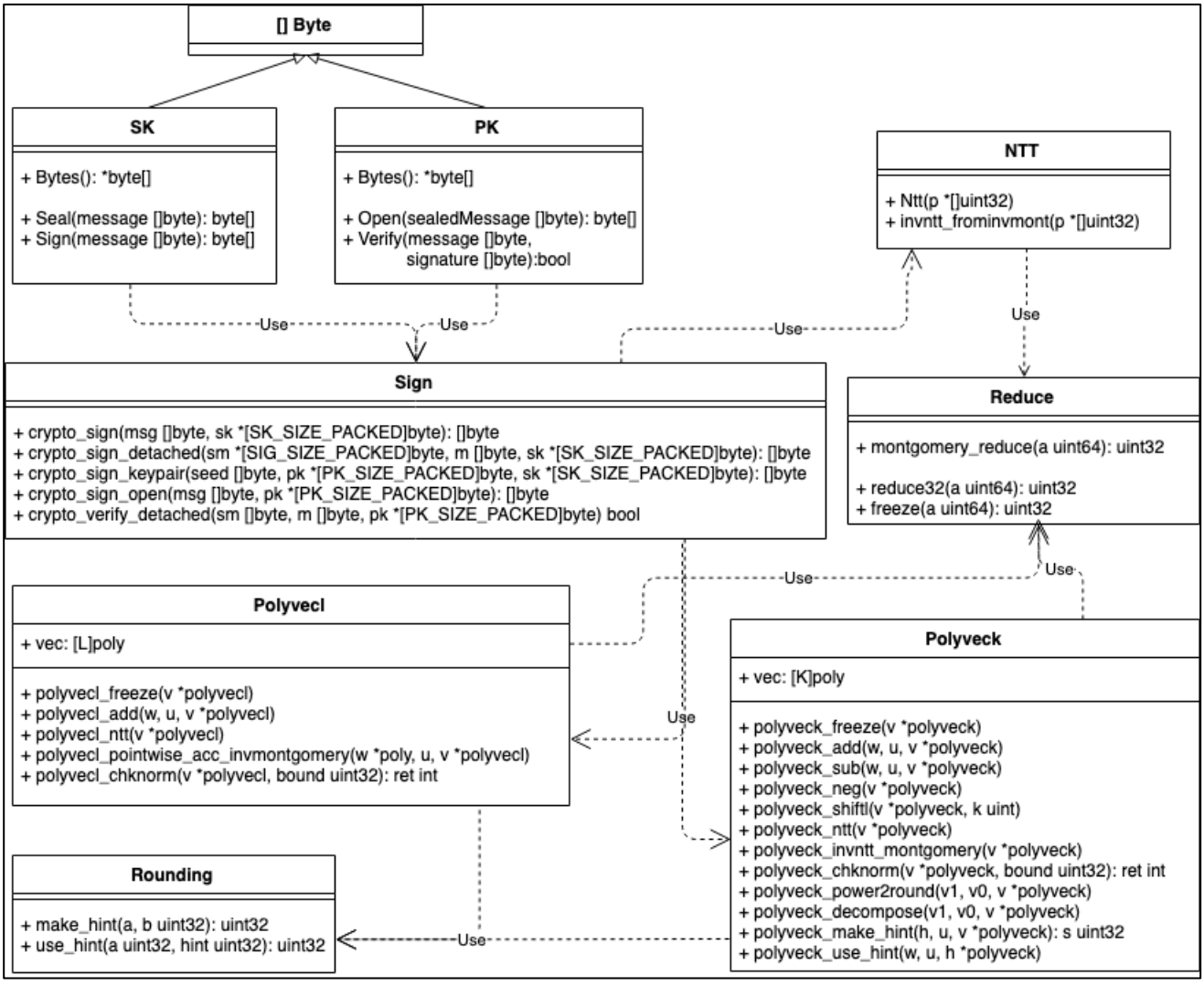

Рисунок 4.3 – Діаграма класів

Реалізація алгоритму – це бібліотека, реалізована на мові Golang. Бібліотека містить два відкритих типи данних: SK – секретний ключ, та PK – публічний ключ. Кожен з ключів є обгорткою над массивом байтів, що являє собою ключ у системі

цифрового підпису Dilithium. SK – тип секретного ключа має наступні методи: Bytes, Sign, Seal.

Bytes – метод не приймає жодні параметри й вертає массив бітів, що складають секретний ключ, створений для зручного представлення секретного ключа для тестування. Sign приймає повідомлення, представлене у масиві байтів, підписує його й повертає підпис, що відповідає цьому повідомленню без самого повідомлення. Підпис має перемінний розмір, але ніколи не перевищує розмір SK_SIZE_PACKED значення, що обчислюється в залежності від параметрів вказаного рівня безпеки. Метод Seal підписує повідомлення й повертає масив байтів, що включає в себе саме повідомлення й підпис. PK – тип секретного ключа має наступні методи: Bytes, Verify, Open. Bytes працює так само, як аналогічний метод в SK клас і не приймає жодні параметри й вертає массив бітів, що складають публічний ключ, створений для зручного представлення секретного ключа для тестування. Verify приймає повідомлення та підпис у вигляді масивів байтів, й перевіряє чи валідний підпис. Open метод приймає масив байтів підписанного повідомлення, відкриває повідомлення, перевіряє підпис й повертає початкове повідомлення, якщо підпис валідний, чи nil, якщо ні.

Класи SK та PK є обгорткою, що надає зовнішній API бібліотеки. Основна логіка підписання та перевірки документу реалізована у наборі методі Sign. Так як у Golang мові немає класів як таких, даний набір методів не прив’язаний до певного класу і сгрупований за логікою та файлом. Sign методи реалізовують основну логіку роботи алгоритму й використовуються SK та PK классами. Дані методи використовують методи, що реалізують NTT (Number Theoretic Transform) перетворення, а також класи Polyvecl та Polyveck, що є обгортками над векторами розмірністю L та K – параметрами безпеки, що задаються при запуску додатку. Класи Polyvecl та Polyveck реалізовують методи роботи з векторами.

Клас Rounding реалізовує ключову особливість алгоритму Dillithium: скорочує розмір підпису за рахунок формування «підказки». Даний клас реалізовує два методи make_hint та use_hint. Метод make_hint отримує массив, що є основою для формування ключів, й массив підказок, в котрий для кожного значення

початкового масиву буде розрахована підказка й записана в масив підказок. Також розраховується загальна сума усіх підказок, що необхідна для подальшої перевірки: якщо загальна сума буде більша, ніж значення парамеру безпеки OMEGA, що задається при запуску програми, підпис не буде сгенеровано. Метод use_hint використовується при перевірці підпису, застосовуючи масив hint, що був сгенерований в make_hint методі. Методи, що відносяться до групи Reduce, реалізовують хешування та алгоритм Монтгомері, що дозволяє прискорити виконання операцій множення та приведення до квадрату, що необхідні при зведенні числа в степінь по модулю, коли модуль має числа порядку сотен біт. Послідовність методів та взаємодія з зовнішнім кодом розглянута на рисунку 4.4.

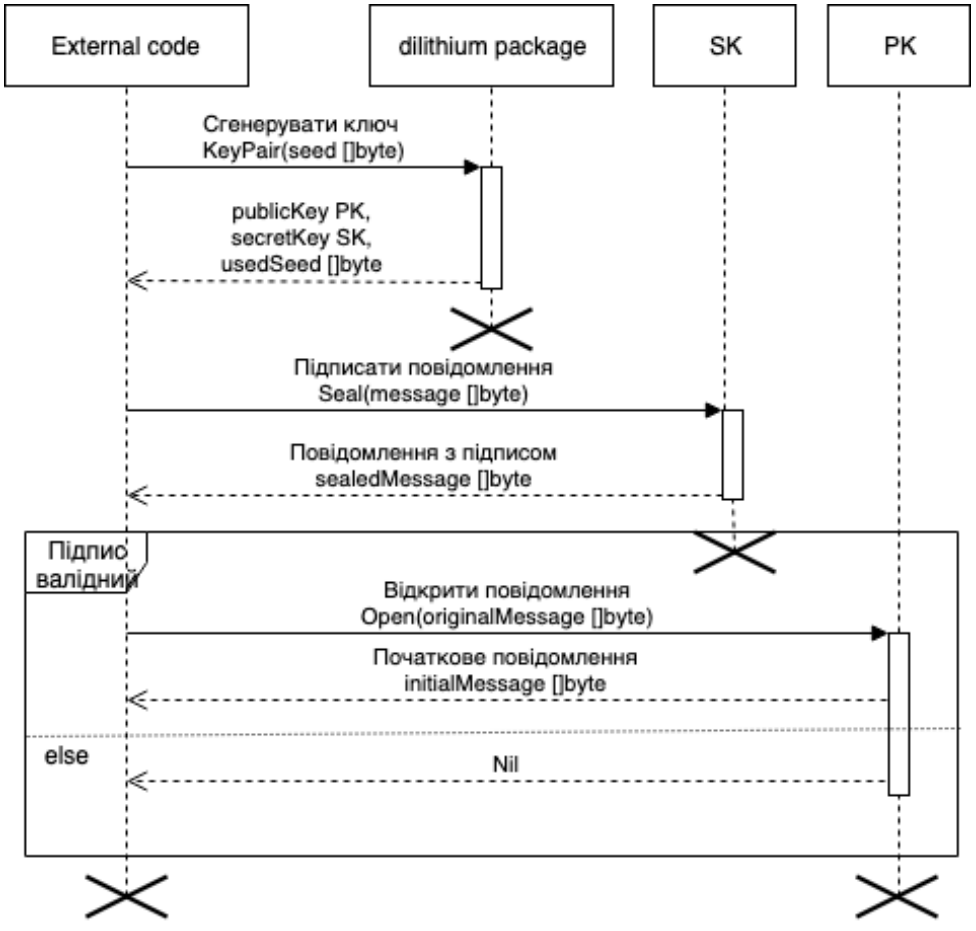

Рисунок 4.4 – Діаграма послідовності

Робота з бібліотекою Dilithium починається з того, що зовнішні код викликає функцію KeyPair з бібліотеки Dilithium. Для генерації ключів була реалізована

функція KeyPair, що приймає масив байтів seed, що є початковими данними для створення секретного та публічного ключа, й повертає секретний та публічний ключ. Якщо в якості параметра буде передано nil, то seed буде створено й заповнено рандомними значеннями.

Далі з використанням секретного ключа, що був сгенерований функцією KeyPair, зовнішній ключ викликає метод Seal класа SK, в котру передає повідомлення у вигляді масиву байтів. Метод повертає повідомлення з підписом також у форматі масиву байтів. Наступним кроком є перевірка підпису. Для цього використовується метод Open, в який передається повідомлення, щ обуло підписане на минулому кроці. Якщо підпис є валідним, то повертається початкове повідомлення у вигляді масиву даних, ящко ні, то повертається Nil. Дані методи реалізовують детерміновану модель алгоритму Dilithium. Лістниг коду з представленними методами наведено у додатку Б.

4.4 Тестування програмного додатку

Тестування програмного забезпечення – це невід’ємною частиною добре поставленого процессу розробки. Тестування забезпечує відповідність функціональних та нефункціональних вимог до реалізованого продукту, а також підвищує якість продукту.

Згідно з принципами піраміди тестування найефективнішими при тестуванні логіки є юніт-тести. Юніт-тести покликані тестувані найменші неподільні частини коду [24]

. За рахунок швидкого виконання та тестування невеликих частин коду – юніт тести є найменш затратними й дозволяють виявити помилки на ранніх етапах розробки, що зменшує вартість помилки. Так як реалізований програмний продукт є бібліотекою й не має інтерфейсу й не взаємодіють з будь-якими сервісами, то такі тести, як end-to-end, інтеграційні та UI тести, є неактуальними. Також необхідно

провести функціональне тестування для перевірки відповідності програми зазначеним функціональним вимогам.

Під час розробки бібліотеки алгоритму Dilithium використовувались юніт-тести, зразок коду текстів представлений на рисунку 4.5. Після завершення роботи було проведено функціональне мануальне тестування, щоб перевірити, що робота додатку реалізовує зазначені функції генерації ключів, підпису та перевірки підпису.

```
run test | debug test
func TestVerify_ShouldPass_IfSignIsValid(t *testing.T) {
    pk, sk, _ := KeyPair(nil)
    msg := []byte("hello")
    sealed := sk.Seal(msg)
    if bytes.Compare(pk.Open(sealed), msg) != 0 {
        t.Fatal("basic signing failed")
    }
}

run test | debug test
func TestSeal_ShouldReturnNil_WhenSignIsInvalid(t *testing.T) {
    pk, sk, _ := KeyPair(nil)
    msg := []byte("hello")
    sealed := sk.Seal(msg)
    sealed[5] ^= 1
    sealed[6] ^= 1
    sealed[7] ^= 1
    if bytes.Compare(pk.Open(sealed), msg) == 0 {
        t.Fatal("broken signature accepted")
    }
}
```

Рисунок 4.5 – Приклад юніт тестів

Результати тестів показали, що розроблена бібліотека працює без збоїв та має працездатний функціонал логіки роботи алгоритму згідно до висунутих вимог та відповідає алгоритму.

ВИСНОВКИ

В ході виконання атестаційної роботи було проведено аналіз сучасних алгоритмів цифрових підписів вразливих та стійких до атак квантовими комп'ютерами, було проаналізовано дослідження у сфері розробки квантових комп'ютерів з точки зору реальної загрози злому алгоритмів цифрового підпису з допомогою атаки квантовим комп'ютером в найближчому майбутньому.

Була досліджена наукова література та найсучасніші дослідження у сфері пост-квантової криптографії. Досліджено стан пост-квантової криптографії, напрямки розвитку, а також потенційні пост-квантові алгоритми цифрового підпису, запропоновані NIST як стійкі до квантових атак схеми цифрових підписів, а саме Picnic, Rainbow, MQDSS, LUOV, GeMSS, SPHINCS+, qTesla, FALCON, CRYSTAL-DILITHIUM. Було зроблено порівняльний аналіз алгоритмів за основними критеріями, такими як рівень безпеки, стійкість до атак по часу, довжина підпису публічного ключа, кількість циклів підписання та перевірки, тобто швидкість роботи алгоритму.

При виконанні атестаційної роботи було у результаті порівняльного аналізу було виявлено, що найкращий за обраними критеріями є алгоритм цифрового підпису Dilithium. Було детально проаналізовано й досліджено особливості реалізації алгоритму Dilithium. У результаті атестаційної роботи було реалізовано алгоритм Dilithium на мові програмування Golang з використанням середовища розробки – Visual Studio Code. Було проведено юніт-тестування, що забезпечує коректність роботи алгоритму. Отримані результати дослідження можуть бути використані при необхідності вибору алгоритму цифрового підпису для практичного використання, а реалізований алгоритм може використовуватися як бібліотека для інших мов програмування при написанні додатків, що сумісні з мовою програмування Golang.

ПЕРЕЛІК ПОСИЛАНЬ

1. Shor P. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer // SIAM J. Comput: електрон. версія, 1997. № 26 (5). С. 1484–1509. URL: https://epubs.siam.org/doi/10.1137/S0097539795293172 (дата звернення: 21.02.2019).

2. Preskill J. Reliable Quantum Computers // The Royal Society: електрон. версія, 1998. № A454. С: 385–410. URL: http://dx.doi.org/10.1098/rspa.1998.0167 (дата звернення: 21.02.2019).

3. Lidar D., Brun T., eds., Quantum Error Correction // Cambridge University Press: електрон. версія, 2013. – 666с. URL: http://dx.doi.org/10.1017/cbo9781139034807 (дата звернення: 22.02.2019).

4. Barends R., Kelly J., Megrant A., eds., Superconducting quantum circuits at the surface code threshold for fault tolerance // Nature: електрон. версія, 2014. № 508 (7497). С. 500–503. URL: http://dx.doi.org/10.1038/nature13171 (дата звернення: 22.02.2019).

5. Harty T.P., Allcock D.T.C., eds, High-Fidelity Preparation, Gates, Memory, and Readout of a Trapped-Ion Quantum Bit // Phys. Rev. Lett.: електрон. версія, 2014. № 113 (22. URL: http://dx.doi.org/10.1103/PhysRevLett.113.220501. (дата звернення: 22.02.2019).

6. National Quantum Initiative Signed into Law [Електронний ресурс] // American Institute of Physics. URL: https://www.aip.org/fyi/2019/national-quantum-initiative-signed-law (дата звернення: 24.02.2019).

7. Горбенко І.Г., Ганзя Р.С. Аналіз можливостей квантових комп’ютерів та квантових обчислень для криптоаналізу сучасних криптосистем // Східно-Європейський журнал передових технологій. 2014. № 19(67). С.:8-16.

8. NISTIR 8105. Report on Post-Quantum Cryptography [Електронний ресурс] // NIST. URL: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf (дата звернення 29.02.2019)

9. Гайнутдинова А. Ф. Квантовые вычисления [Електронний ресурс]: метод.пособие/Казанский гос. ун-т, 2009, – 272с. URL: https://www.twirpx.com/file/2127815/ (дата звернення 24.02.2019).

10. Proos J. Shor's discrete logarithm quantum algorithm for elliptic curves // QIC: електронна версія. 2003. № 4. С.: 317-344.URL: https://dl.acm.org/citation.cfm?id=2011531 (дата звернення 26.02.2019).

11. Гармаш Д.В., Баликов О.О., Філатова Н.В. Квантові криптографічні алгоритми електронного підпису на основі мультиваріативних квадратичних перетворень // Прикладна радіотехніка. 2016. № 15 (3). С.:215-225.

12. Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms [Електронний ресурс] // Federal Register Notice. №81. С.: 92787-92788. Дата оновлення: 20.12.2016. URL: https://federalregister.gov/a/2016-30615 (дата звернення 07.03.2019).

13. Горбенко І.Г., Кузнєцов О.О. Постквантова криптографія та механізми її реалізації // Радіотехніка. 2016. № 186. С.: 32-52.

14. Galbraith S.D. Constructing isogenies between elliptic curves over Finite Fields // London Mathematical Society: електрон. версія. 1999. № 2: С.: 118–138. URL: https://www.cambridge.org/core/journals/lms-journal-of-computation-and-mathematics/article/constructing-isogenies-between-elliptic-curves-over-finite-fields/30811E8101E76F7C2A22873EE7080237 (дата звернення 17.03.2019).

15. NISTIR 8240. Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process [Електронний ресурс] // NIST. URL: https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf (дата звернення 27.03.2019)

16. Ducas L., Kiltz E., eds. CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation [Електронний ресурс] // CRYSTALS. URL: https://pq-crystals.org/dilithium/ (дата звернення 14.04.2019)

17. Орлов А. Теория принятия решений — Москва: Март, 2004 — 656 с.

18. Ногин В. Сужение множества Парето: аксиоматический подход — Москва: ФИЗМАТЛИТ, 2015 — 236 с.

19. Kiltz E., Lyubashevsky V., Schaffner Ch. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle // Cryptology ePrint: електрон. версія. 2017. № 916. URL: https://eprint.iacr.org/2017/916 (дата звернення 26.04.2019)

20. Lyubashevsky V. Fiat-Shamir with aborts: Applications to lattice andfactoring-based signatures [Електроний ресурс] // Research Gate. URL: https://www.researchgate.net/publication/221326768_Fiat-Shamir_with_Aborts_Applications_to_Lattice_and_Factoring-Based_Signatures (дата звернення: 26.04.2019).

21. Güneysu T., Lyubashevsky V., Pöppelmann Th. Practical lattice-based cryptography: A signature scheme for embedded systems // Ches, електрон. версія. 2012. № 2: С.:530-547. URL: Th. Practical lattice-based cryptography: A signature scheme for embedded systems (дата звернення 27.04.2019)

22. Samwel N., Batina L., eds. Breaking ed25519 in wolfssl // Cryptology ePrint: електрон. версія. 2017. № 985. URL: https://eprint.iacr.org/2017/985 (дата звернення 28.04.2019)

23. The Go Programming Language [Електроний ресурс] // The Go Programming Language. URL: https://golang.org/ (дата звернення: 01.05.2019).

24. Osherove R. The Art of Unit Testing— NY: Manning, 2013 — 292 с.