Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерної інженерії та управління _____
(повна назва)

Кафедра _____Безпеки інформаційних технологій _____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

___Постквантові алгоритми цифрового підпису та умови їх реалізації_____
(тема)

Виконав:

студент __2_ курсу, групи __БІКСм-20-1_____

____Чумак В.І._____

(прізвище, ініціали)

Спеціальність _____125 Кібербезпека _____

(код і повна назва спеціальності)

Освітня програма ___«Безпека інформаційних
_____ і комунікаційних систем»
__ (повна назва освітньої програми)

Керівник __доцент Петренко О. Є._____

(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ _____Халімов Г.З.__
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет_____Комп'ютерної інженерії та управління _____

Кафедра_____Безпеки інформаційних технологій_____

Рівень вищої освіти_____ другий (магістерський)_____

Спеціальність_____125 Кібербезпека _____
(код і повна назва)

Освітня програма_____«Безпека інформаційних і комунікаційних систем»___
(повна назва)

ЗАТВЕРДЖУЮ:

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові_____Чумаку В'ячеславу Ігоровичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи ____Постквантові алгоритми цифрового підпису та умови їх реалізації
_____
затверджена наказом по університету від «08» листопада 2021 р. № 1685Ст
2. Термін подання студентом роботи до екзаменаційної комісії 14.12.2021 р.
3. Вихідні дані до роботи _ статистичні дані щодо постквантових алгоритмів цифрового підпису_____
_____ _____
_____
_____
_____
_____
_____
_____
_____
4. Перелік питань, що потрібно опрацювати в роботі _____
Досягнення у створенні квантових комп'ютерів.
Опис постквантових алгоритмів основаних на решітках.
Порівняльний аналіз постквантових алгоритмів формування цифрових підписів
Умови реалізації постквантових алгоритмів цифрового підпису.
Основні атаки на постквантові алгоритми цифрового підпису.
_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) _презентаційний матеріал у вигляді слайдів _____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

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	Вибір здобувачем теми кваліфікаційної роботи	02.09.2021	Виконано
2	Затвердження плану і завдання кваліфікаційної роботи	09.11.2021	Виконано
3	Аналіз завдання, пошук та аналіз літературних джерел за темою роботи	10.11.2021-18.11.2021	Виконано
4	Виконання кваліфікаційної роботи	19.11.2021-30.11.2021	Виконано
5	Оформлення пояснювальної записки	01.12.2021-12.12.2021	Виконано
6	Здача на перевірку та підпис кваліфікаційної роботи керівнику	13.12.2021	Виконано
7	Проходження перевірки на плагіат та нормоконтроль кваліфікаційної роботи	15.12.2021	Виконано
8	Допуск завідувачем кафедри до захисту кваліфікаційної роботи	15.12.2021	Виконано
9	Захист кваліфікаційної роботи	17.12.2021	Виконано

Дата видачі завдання _____ _____20___ р.

Студент _____
(підпис)

Керівник роботи _____ _____ доцент Петренко О.Є._
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи містить : 80 с., 3 табл., 9 рис., 25 джерела.

ПОСТКВАНТОВА КРИПТОГРАФІЯ, АЛГОРИТМ ГРОВЕРА, АЛГОРИТМ ШОРА, КВАНТОВИЙ КОМП'ЮТЕР, КВАНТОВИЙ КРИПТОАНАЛІЗ, NTRU.

Об'єкт дослідження – реалізація постквантових алгоритмів цифрового підпису.

Предмет дослідження – постквантові алгоритми цифрового підпису.

Мета роботи – обґрунтувати вибір параметрів та умов для реалізації посткавнтового алгоритму побудови цифрового підпису та шифрування.

Методи дослідження – аналіз та синтез інформації щодо постквантових алгоритмів цифрового підпису, метод попарних порівнянь.

ABSTRACT

Explanatory note to the qualification work contains: 80 pages, 3 tables, 9 figures, 25 sources.

POSTQUANTUM CRYPTOGRAPHY, GROWER ALGORITHM, SHORE ALGORITHM, QUANTUM COMPUTER, QUANTUM CRYPTOANALYSIS, NTRU.

The object of research is the implementation of post-quantum digital signature algorithms.

The subject of research is post-quantum digital signature algorithms.

The purpose of the work is to substantiate the choice of parameters and conditions for the implementation of the post-quantum algorithm for building a digital signature and encryption.

Research methods - analysis and synthesis of information on post-quantum digital signature algorithms, method of pairwise comparisons.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

GSA – Grover search algorithm

RSA – Rivest, Shamir і Adleman

ECC – Error-correcting code memory

NTRU – Nth-degree TRUncated polynomial ring

CNOT – controlled-NOT

NTRUEncrypt – Nth-degree TRUncated polynomial ring

NIST – National Institute of Standards and Technology

KEM – key-encapsulation mechanism

BKW – Blum, Kalai, Wasserman

LPN – Learning parity with noise

LWE – Learning with errors

BDD – Bounded distance decoding

ЕЦП – Електронний цифровий підпис

UOV - Unbalanced Oil and Vinegar

ВСТУП

У сучасному світі інформація є одним з найбільш цінних ресурсів, а інформаційні технології стрімко розвиваються. Тому в теорії інформації на перший план виходить захист даних при зберіганні та передачі іншим особам. Це завдання не можна віднести до новим завданням теорії інформації: з'явилася вона не з повсюдним поширенням комп'ютерних технологій, а набагато раніше. Завдання шифрування інформації для безпечного зберігання та безпечної передачі даних виникло близько 4 тис. років тому. В наші дні людство суттєво просунулося у цій сфері, проте на цьому розвиток не зупиняється – паралельно із удосконаленням захисту даних удосконалюються і методи атак на її злому.

Обсяг даних, що створювалися щодня, просто величезний, і сучасні комп'ютери вже не завжди встигають за такими обсягами. Сучасні суперкомп'ютери, як і раніше, занадто повільні для виконання деяких найважливіших наукових завдань, наприклад, тестування впливу нових лікарських препаратів на молекулярному рівні.

Завдяки можливості виконувати дуже складні обчислення значно швидше, або навіть моделювати ці ліки на молекулярному рівні, квантові комп'ютери здатні надати таке необхідне зростання продуктивності та швидкості. Більшість фахівців погоджуються з тим, що квантові комп'ютери – це наш шанс впоратися з викликами 21 століття.

В даний час в технологічному світі, що швидко розвивається, поява квантового комп'ютера все більше стає реальністю. Для деяких областей знань створення такого пристрою дозволить обробляти дані набагато швидше, ніж це роблять сучасні машини, що стане проривом, але для сучасної криптографії - загрозою злому всіх існуючих криптосистем.

Провідними світовими науковими колективами постійно ведуться розробки щодо побудови квантового обчислювача. Деякі вчені дають тимчасову оцінку 20 років реалізації повномасштабного квантового комп'ютера. Якщо обчислювальні можливості порушника зростуть у десятки, сотні, тисячі разів, це призведе до різкої необхідності збільшення довжини ключів до критичного рівня, не придатного їх успішної експлуатації реальних інформаційних системах. Також, необхідно відзначити, що з появою квантового супротивника з величезними обчислювальними потужностями велика ймовірність і тотального злому існуючих криптосистем шляхом повного перебору по всьому простору ключів.

Цю проблему можна вирішити шляхом використання примітивів постквантової криптографії, порівняно нової галузі криптографії, покликаної протистояти квантовим обчисленням.

1 ЗАГАЛЬНА ІНФОРМАЦІЯ ПРО КВАНТОВІ КОМП’ЮТЕРИ

1.1 Квантові комп’ютери

Квантовий комп'ютер — засіб обчислювальної техніки, де за основу роботи центрального процесора лежать закони квантової механіки. Такий комп'ютер принципово відрізняється від традиційних ПК, що працюють на основі кремнієвих чіпів. Поки що квантовий комп'ютер - пристрій, про який говорять багато дослідників фізики обчислень.

Цей пристрій застосовується для обчислення не класичних алгоритмів, а процеси квантової природи це квантові алгоритми, що використовують ефекти квантової механіки, такі як квантовий паралелізм і квантова заплутаність.

На даний момент універсальний квантовий комп'ютер залишається гіпотетичним пристроєм, проте вже реалізовані поодинокі експериментальні системи, які виконують фіксований алгоритм невеликої складності. Одним із таких комп'ютерів є адіабатичний квантовий комп'ютер D-Wave, права на який зараз належать компанії Google. Остання реалізація адіабатичного комп'ютера D-Wave – D-Wave 2000q – містить 2000 кубіт (проте не всі є попарно заплутаними, загальний набір розпадається на безліч регістрів до 8 попарно заплутаних кубітів (див. рис. 1.1).

Цей комп'ютер працює за принципом квантового відпалу, або квантової нормалізації, і є системою з великої кількості компонентів і керуючих параметрів. Процес квантового відпалу полягає в пошуку глобального мінімуму за допомогою заміни поточного рішення на сусіднє випадковим чином, якщо в сусідньому рішенні функціонал, що оптимізується, менше. Цей процес регулюється параметром "напруженість поля тунелювання", який при старті досить великий, тому пошук проводиться по всьому просторі. При

зменшенні поля напруженості система «осідає» в кількох станах з мінімальними значеннями функціоналу, що оптимізується, одне з яких відповідатиме глобальному мінімуму. У межі виходить класична система одному з основних станів.

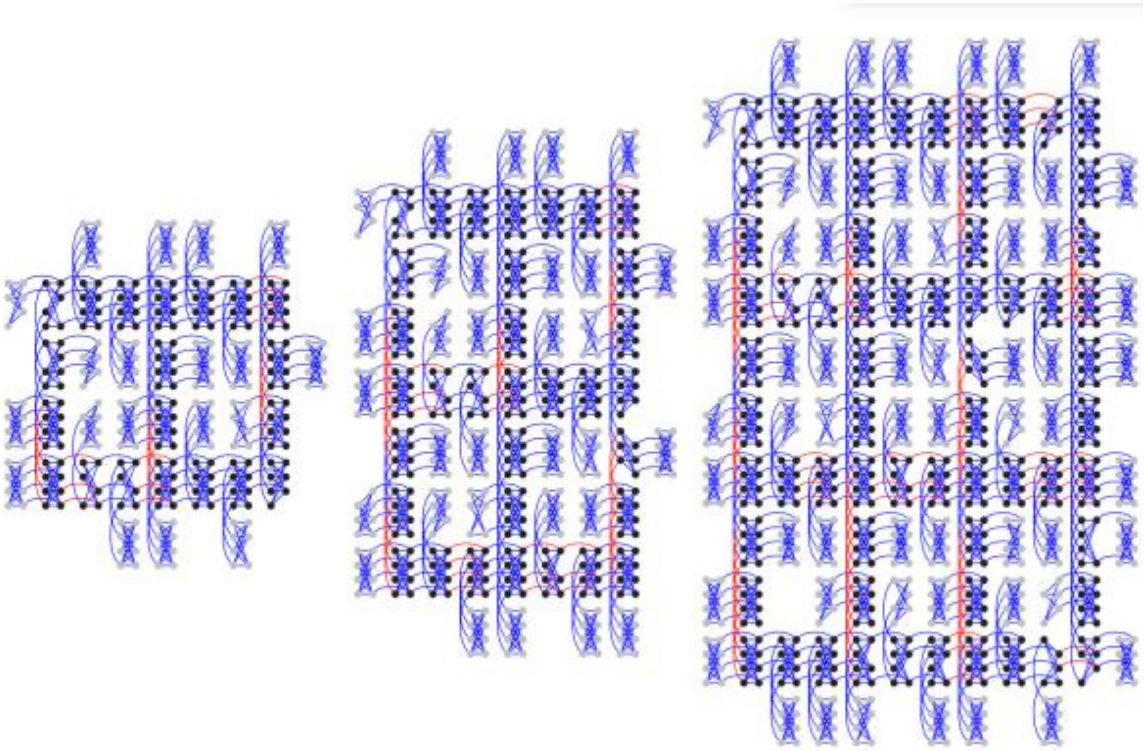

Рис 1.1 Загальна схема заплутаних кубітів комп'ютерів D-Wave

Квантовий комп'ютер D-Wave працює за низьких температур і повільно змінює свій стан за рахунок квантового тунелювання при зміні заданих параметрів. На жаль, цей комп'ютер здатний вирішувати обмежений клас задач оптимізації та не підходить для реалізації класичних квантових алгоритмів та квантових вентилів.

Базою для обчислень такого типу служить кубіт - система, у якій число частинок аналогічне імпульсу, а фазова змінна (енергетичний стан) - координаті. Фазовий кубіт був уперше реалізований у лабораторії Делфтського університету (Нідерланди) і з того часу активно вивчається.

На відміну від звичайного біта, здатного мати тільки значення 1 і 0, квантовий біт (кубіт) може перебувати в суперпозиції цих станів, тобто

одночасно в значенні 1 і 0. На практиці кубіт може існувати в різних комбінаціях цих значень, що в перспективі дозволить створювати надшвидкі комп'ютери. Кубіти стануть будівельними елементами майбутніх квантових комп'ютерів, здатних вирішувати завдання, практично недоступні класичним цифровим комп'ютерам. Для виконання обчислень на квантовому комп'ютері необхідна взаємодія кількох кубітів, причому таким чином, щоб вони утворили єдину квантову систему. Потім цій системі треба дозволити розвиватися за законами квантової механіки і через певний час з'ясувати, який стан вона прийшла.

Зі зростанням числа об'єднаних кубітів, обчислювальна потужність такої квантової системи експоненційно зростає. Теоретично це дозволяє квантовому комп'ютеру справлятися із завданнями, на які звичайному цифровому комп'ютеру знадобляться мільйони років. Наприклад, давно відомий алгоритм Шора, що дозволяє швидко розкладати великі числа на прості множники (завдання, необхідне для злому сучасних шифрів). Звичайні комп'ютери вирішують це перебором можливих дільників, тому довгі числа сучасні комп'ютери можуть обробляти роками. Квантовий комп'ютер впорався б із таким завданням за лічені хвилини і навіть секунди, залежно від продуктивності.

1.1.1 Біт і Кубіт

В основі класичної теорії інформації лежить поняття біта. Біт є одиницю виміру інформації в двійковій системі числення, назва утворена від англійського словосполучення binary digit - двійкове число. В якості фізичної реалізації найчастіше використовується тригер, який може перебувати лише у двох стійких станах (нульовий та високий). Інакше біт можна інтерпретувати як електричний розряд (0 за його відсутності та 1 за його наявності).

В основі квантової теорії інформації біт замінюється на кубит (quantum bit – квантовий біт) і є квантовим розрядом або найменшим елементом для зберігання інформації в квантовому комп'ютері. Як і біт, кубіт має два власні стани – $|0\rangle$ і $|1\rangle$ (В позначеннях Дірака). Основна відмінність кубіту від біта полягає в тому, що кубіт може бути у суперпозиції цих станів, тобто. в стані $\alpha|0\rangle + \beta|1\rangle$, де α і β – комплексні числа, що задовольняють умові $|\alpha|^2 + |\beta|^2 = 1$. Термін qubit був запроваджений С. Візнером у 1983 р. При вимірі кубіту його стан руйнується, в результаті буде отримано або стан $|0\rangle$ з ймовірністю $|\alpha|^2$, або стан $|0\rangle$ з ймовірністю $|\beta|^2$ так що геометрично кубіт можна інтерпретувати як одиничний вектор у двовимірному комплексному просторі, а квантові оператори – як дія унітарних матриць на вектор.

Фейнман, який породив квантову теорію інформації своєю промовою на конференції «Physics of Computation» у Массачусетському Технологічному інституті, пропонував як фізичну реалізацію кубіту фотони, а як можливі стани кубіту – вертикальну та горизонтальну поляризацію фотонів. Надалі були запропоновані інші реалізації кубітів, такі, як спини ядер та електронів, квантові точки, NV-центри в алмазах, захоплені іони та атоми та інше.

1.1.2 Квантові обчислення та схеми

Квантові обчислення – мова для опису зміни квантового стану. Класичний комп'ютер будується з електричних схем, що містять дроти та логічні елементи, що дозволяє передавати класичну інформацію та керувати нею. Квантовий комп'ютер зручно представляти аналогічно класичному комп'ютеру у вигляді квантових схем з дротів та квантових елементів, які дозволяють передавати квантову інформацію та керувати їй.

Наприклад розглянемо одне із найважливіших для квантових схем оператор CNOT (controlled-NOT). Це найпростіший елемент на двох кубітах,

один із яких є керуючим, а другий – керованим. Його умовне позначення представлено на рис. 1.2 де верхня лінія являє собою керуючий кубіт, а нижня - керований.

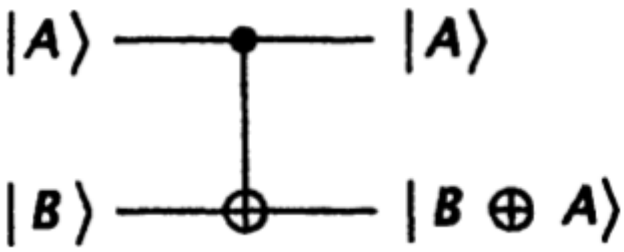

Рис. 1.2 Умовне позначення елемента CNOT

Якщо керуючий кубіт встановлений 1, то керований кубіт змінює свій стан, якщо керуючий кубіт встановлено 0, то значення керованого кубіту не змінюється. Формальний запис виглядає так:

$$|00\rangle \rightarrow |00\rangle,\ |01\rangle \rightarrow |01\rangle,\ |10\rangle \rightarrow |11\rangle,\ |11 \rightarrow |10\rangle.$$

Інакше елемент CNOT можна подати як узагальнення класичного елемента XOR та описати його дію як $|A, B\rangle \rightarrow |A, A \oplus B\rangle$, де операція $\oplus$ являє собою додавання за модулем 2 (те, що виконує логічний оператор (XOR).

Інший спосіб описати елемент CNOT – дати його подання у матричній формі для впливу на вектор ймовірності виду $(\alpha_1, \beta_1, \alpha_2, \beta_2)^T$, де α_1, β_1 описують стан керуючого кубіту, а α_2, β_2 - стан керованого кубіту. Матрична форма елемента CNOT, яка записана для порядку амплітуд $|00\rangle$, $|01\rangle$, $|10\rangle$ і $|11\rangle$:

$$U_{CN} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

U_{CN} є унітарною матрицею, тому вимога збереження ймовірності виконується.

1.2 Стан досягнень у побудові квантових комп’ютерів

Побудовою квантового комп’ютера займається багато наукових колективів та дослідних груп по усьому світу, але значних результатів у цьому напрямку, на даний момент, як свідчать данні досягли тільки декілька колективів з США.

Канадська компанія D-Wave оголосила про створення машини для квантового відпалу на 5000 кубітах, який перевершує минуле покоління пристроїв за розміром, кількістю зв'язків між кубітами та швидкістю роботи.

У той час як такі компанії як Google, IBM та Rigetti намагаються побудувати повноцінний квантовий комп'ютер, канадська компанія D-Wave вже кілька десятиліть йде іншим шляхом. D-Wave працює над створенням надпровідних квантових адіабатичних обчислювачів на основі квантового відпалу - це пристрої, які можуть вирішувати лише завдання оптимізації, але завдяки квантовому ефекту, тунелюванню, потенційно вони здатні впоратися з дуже великими завданнями, недоступними для класичних пристроїв.

Суть квантового відпалу полягає в тому, що квантова система при досить низькій температурі знаходиться в основному стані з найменшою енергією. Це найменша енергія визначається гамільтоніаном (функцією енергії), який задається видом досліджуваної функції - у цієї функції шукається мінімум. Якщо відпалювач перебуває в основному стані з якимось початковим гамільтоніаном, то плавно приводячи гамільтоніан до бажаного виду, ви весь час залишаєтеся в основному стані. Цей ефект забезпечується адіабатичною теоремою. Таким чином, кінцевий стан відповідає мінімальній енергії кінцевого гамільтоніану і є мінімумом функції.

Переведення одного гамільтоніана в інший здійснює за допомогою зміни магнітних полів, які діють на надпровідні кубити та їх зв'язки з сусідами. Чим більше кубитів і більше зв'язків, тим більше контролю, тож можна вирішувати складніші завдання. Проблема сучасних пристроїв у тому, що

поки що вони надто маленькі, щоб конкурувати з класичними методами оптимізації. Минуле покоління D-Wave 2000Q мало 2000 кубітів і кожен кубит був пов'язаний із шістьма сусідами, всього близько 6000 зв'язків. Така топологія дуже обмежувала розмір завдання, яку можна вирішувати на цьому пристрої.

D-Wave представили нове покоління квантових адіабатичних обчислювачів D-Wave Advantage с 5000 кубитами, де кожен кубит пов'язаний із 15 сусідами. Загалом у такій машині понад 35000 зв'язків, що у 6 разів більше, ніж у попередньому поколінні. Така топологія отримала ім'я Pegasus та є важливим інженерним досягненням, яке може бути використане і для універсальних квантових комп'ютерів у майбутньому. Компанія стверджує, що за допомогою цієї машини можна вирішити завдання з більш ніж мільйоном змінних.

Вчені з D-Wave також додали класичні методи передпостобробки, які прискорили процес вирішення оптимізаційних завдань, на 64% в порівнянні з минулим поколінням обчислювачів. Компанія D-Wave є лідером на ринку квантових відпалювачів, у 2017 році компанія анонсувала старт продажу своїх машин.

Також Корпорація IBM представила Q System One – компактний модульний квантовий комп'ютер, який самі представники компанії назвали “першою у світі інтегрованою універсальною квантовою обчислювальною системою, розробленою для наукового та комерційного застосування”.

Анонсована на міжнародній виставці споживчої електроніки CES 2019 система є 20-кубітним обчислювальним пристроєм четвертого покоління, укладеним у герметичний корпус у формі куба з гранню довжиною 2,75 м, який виконаний з боросилікатного скла товщиною 1,27 см. Крім квантового процесу Q System One розташовуються різні модулі, що управляють, а також система охолодження.

Вибір матеріалу корпусу, за даними ресурсу Engadget, обґрунтований простотою підтримки необхідної для функціонування пристрою температури – близько 10 мілікельвінів, тобто досить близько до абсолютного нуля. Крім того, конструкція корпусу дозволяє оберігати компоненти квантового комп'ютера від небажаних вібрацій, які можуть призвести до виникнення обчислювальних помилок у процесі його роботи.

"Об'єднання всього цього в перший інтегрований квантовий комп'ютер загального призначення є знаковим моментом для IBM з огляду на її історію", - цитує Даріо Джила (Dario Gil), головного операційного директора IBM Research, видання Financial Times, протиставляючи новинку лабораторним зразкам квантових обчислювальних систем минулих років. , які часом займали цілі кімнати.

Незважаючи на відносну компактність новинки, продемонструвати повнорозмірний працюючий прототип в IBM не наважилися - замість нього погляду публіки зменшилася до 2,25 м версія Q System One з відсутньою задньою панеллю, за якою ховаються модулі охолодження, живлення та управління, а також без захисного ковпака», під яким ховається власне квантовий комп'ютер. Останнє зроблено для зручності відвідувачів виставки, дозволяючи розглянути всі компоненти пристрою.

Проте, за словами Боба Сатора (Bob Sutor), віце-президента IBM Research, IBM має «повністю працездатний» прототип Q System One в Йорктаун Хайтс (Нью-Йорк, США), який вже застосовується при проведенні експериментів.

IBM Research серйозно просунулися у своїх дослідженнях із створення першого квантового комп'ютера. Але не лише вони беруть участь у «квантових перегонах». Google та Intel також розробляють власні рішення. Ще один гравець у цій сфері – стартап Rigetti із Сан-Франциско. А канадська компанія

D-Wave вже продає комп'ютери, засновані на технології квантового відпалу, які купили НАСА і Google.

У майбутньому квантові комп'ютери обіцяють стати головною обчислювальною силою у вирішенні низки проблем, наприклад задач оптимізації або хімічних симуляціях. Очікується, що квантові обчислення дозволять створювати нові типи ліків та матеріалів, а також швидко знаходити рішення для задачі комівояжера та завдання про максимальний розріз.

1.3 Алгоритми Гровера та Шора. Умови їх реалізації

Алгоритм Гровера був запропонований у 1996 р. у розв'язанні задачі перебору за допомогою оракула – чорної скриньки. У безперервному випадку використовується гамільтоніан квантової системи або адіабатична зміна стану $|0\rangle$. Один з варіантів безперервного алгоритму Гровера було запропоновано 2000 р.

Схема алгоритму Гровера у дискретних змінних наведена на рис. 4. Алгоритм полягає у пошуку рішення рівняння $f(x) = 1$, де f є булева функція від n змінних. Класичні алгоритми пошуку вимагають прямого перебору всіх $N = 2^n$ варіантів, у той час як використання квантового паралелізму та імовірнісного характеру квантових обчислень скорочує кількість варіантів, що перебираються до $\frac{\pi}{4}\sqrt{N}$.

Нехай I_a- Унітарний оператор, що дзеркально відображає гільбертове простір щодо гіперплощини, перпендикулярної вектору a, $|x_{tar}\rangle$ - стан, що відповідає кореню рівняння $f(x) = 1$, а стан $\left|\tilde{O}\right\rangle = \frac{1}{\sqrt{N}} \sum_j |\mathrm{j}\rangle$ рівномірна суперпозиція всіх станів. Тоді алгоритм пошуку Гровера полягає у

застосуванні оператора $G = -I_{\tilde{0}} I_{x_{tar}}$ до стану число разів, що дорівнює цілій частині $\frac{\pi}{4}\sqrt{N}$ Результат буде близьким до стану $|x_{tar}\rangle$.

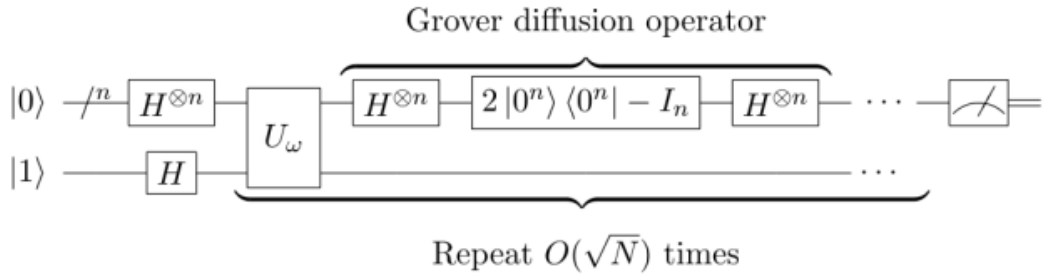

Рис. 1.3 Квантова схема дискретного алгоритму Гровера.

Алгоритм Шора. На нинішній 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 [2, 14]. 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:

$$O(\exp((c + 0(1))(\ln n)^{1/3} (\ln \ln n)^{2/3}.$$

В той же час квантовий алгоритм Шора, має поліноміальну складність. Він здатен розкласти складене число на прості множники приблизно за такий час $O(n^3)$ та з використанням O(n) кубітів. Розглянемо його детальніше.

Поліноміальний за часом алгоритм Шора, що запропонований ним 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 дозволяє розкласти число N з часовою складністю $O(\log^2 N \log^3 (\log N))$ з використанням O (log N) логічних кубітів.

Таким чином, значимість алгоритму полягає в тому, що при використанні квантового комп'ютера з декількома сотнями логічних кубітів він дозволяє, наприклад, зламати RSA криптоперетворення, розклавши модуль перетворення N, тобто знайти множники модуля N. Уже при $N \geq 21^{024}$ 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.

1.4 Пошуки розв’язання проблеми, постквантова криптографія

Створення квантових комп'ютерів відкриє нові можливості для людства, але при цьому існуючі методи захисту інформації втратять свою ефективність. Незважаючи на те, що поки що квантові комп'ютери тільки виходять за межі лабораторій, потреба у використанні квантово-безпечної, або, як її ще називають, постквантової криптографії є вже сьогодні.

Класична криптографія.

Криптографія вивчає методи захисту конфіденційності та цілісності інформації, а також процедури автентифікації. Класична криптографія спирається на математичні алгоритми. Дані, які користувачі передають електронною поштою, через хмарні сервіси або месенджери, захищені тим чи іншим криптографічним протоколом. Кожен з цих протоколів включає набір алгоритмів шифрування, тобто зведення правил, визначальних, як саме буде перетворена інформація.

Алгоритми шифрування для даних у цифровому вигляді можна розділити на два основні типи: симетричні (наприклад, AES) та асиметричні (найпоширеніший – RSA). При симетричному шифруванні сторони, що обмінюються даними, використовують один і той же секретний ключ для шифрування та розшифровування даних. Цей ключ не відомий третій стороні, відповідно вона не має доступу до даних. Симетричні алгоритми зазвичай використовуються для таких завдань, як шифрування великих баз даних, файлових систем та сховищ.

Асиметричні алгоритми передбачають використання двох ключів – відкритого та закритого. Ці ключі пов'язані між собою математичними залежностями. Для шифрування даних використовується відкритий ключ, а для їх дешифрування - закритий ключ з тієї ж пари. Такий різновид асиметричних алгоритмів називається алгоритмами з відкритим ключем.

При симетричному шифруванні сторони повинні мати загальний закритий ключ ще до початку обміну даними, а при асиметричному - починають обмін без загальної секретної інформації: тут одна сторона (одержувач) знає обидва ключі та передає другий (відправнику) тільки відкритий ключ.

Багато криптографічних протоколів є гібридними. Починаючи обмін інформацією, сторони використовують алгоритми з відкритим ключем для передачі одного рядка, а потім переходять на набагато швидші симетричні алгоритми, де цей загальний рядок є секретним ключем.

Криптографія в епоху квантових комп'ютерів.

Класична криптографія на сьогодні надійно забезпечує цілісність та конфіденційність даних. Навіть потужному суперкомп'ютеру знадобляться, мабуть, сотні, а то й тисячі років, щоб вирішити складні математичні завдання, на яких вона базується. Але з появою повномасштабного квантового комп'ютера аналогічне завдання можна буде вирішити за кілька днів або годин. Про це свідчать результати дослідження, яке провів американський учений Пітер Шор у Массачусетському технологічному інституті ще у середині 90-х років XX століття.

Розробку квантових комп'ютерів ведуть дослідницькі групи з усього світу щоб діяти на випередження, але поки що йдеться лише про лабораторні зразки з дуже обмеженою потужністю та функціональністю. Може здатися, що впроваджувати квантово-безпечну криптографію ще рано, але це не так.

Завдяки дослідженням, проведеним міжнародними консорціумами протягом останнього десятиліття, цю проблему вдалося вирішити. На основі ґрат були створені асиметричні схеми шифрування та цифрового підпису з розміром ключа, як у RSA (популярний зараз алгоритм з відкритим ключем), при цьому працює новий алгоритм навіть швидше, ніж класичний. Таким

чином, було доведено, що криптографія на ґратах дозволяє створювати алгоритми, які раніше вважалися нездійсненними.

У 2016 році Національний інститут стандартів та технологій США (NIST) ініціював проект з оцінки та стандартизації одного або кількох квантово-стійких алгоритмів з відкритим ключем. Збір заявок тривав аж до 2018 року. Після першого відбіркового раунду участь у конкурсі продовжили 69 кандидатів, після другого – 26. Ймовірно, вже у 2020 році відбудеться третій раунд, після чого розпочнеться етап розробки стандартів. Планується, що перший проект квантово-безпечного стандарту NIST представить між 2022 та 2024 роками.

Страхування на випадок квантового прориву.

Враховуючи поточні успіхи дослідницьких груп у галузі квантових обчислень, комерційні компанії та уряди країн мають замислитись про цінність своїх даних. Для захисту інформації, яка має залишитися конфіденційною через 10–30 років, впроваджувати квантово-безпечну криптографію рекомендується вже сьогодні, не чекаючи стандартизації.

Важливо розуміти, що перейти на постквантові алгоритми негайно після ухвалення стандартів не вдасться. Знадобиться велика підготовча робота. Нові ключі можуть мати дещо більші розміри, і інфраструктура повинна бути розрахована для їхньої передачі без втрати звичної швидкості комунікації.

У той же час багато пропозицій, поданих у NIST, це лише незначні зміни добре вивчених завдань. Можна вибрати одне або кілька з них та використовувати у тандемі з поточною криптографією. Такий модульний перехід до квантово-безпечної криптографії виглядає найкращим рішенням, тому що він гарантує поточну безпеку даних, навіть у тому випадку, якщо в постквантових алгоритмах виявляться «хвороби молодості», пов'язані з їх використанням без універсальних квантових комп'ютерів. При цьому, впровадивши елементи квантово-безпечної криптографії сьогодні, власники

даних застрахують себе на випадок майбутнього прориву в квантових обчисленнях.

2 ПОСТКВАНТОВІ АЛГОРИТМИ ЦИФРОВОГО ПІДПИСУ ТА ЇХ СПЕЦІФІКА ЗАСТОСУВАННЯ

2.1 Умови конкурсу NICT на кращій алгоритми асиметричного шифрування в постквантовий період

У 1994 році американський вчений П. Шор (P. W. Shor) запропонував квантовий алгоритм, здатний вирішувати завдання факторизації та дискретного логарифмування за поліноміальний час. Це означає, що криптографічні механізми, засновані на перерахованих вище завданнях, втратять свою криптостійкість. До таких механізмів відносяться і чинні стандарти інформаційних технологій. Криптографічний захист інформації. Процеси формування та перевірки електронного цифрового підпису та чинний стандарт Сполучених Штатів Америки ECDSA, та багато інших протоколів [10].

Існує ще один квантовий алгоритм - алгоритм Л. Гровера (L.Grover), він не дає настільки великий виграш у швидкості в порівнянні з алгоритмом Шора і більше підходить для злому симетричних криптосистем та хеш-функцій. Алгоритм Гровера квадратично підвищує швидкість обчислень наперевагу класичним комп'ютерам.

Незважаючи на те, що вже відбувається перехід від хеш алгоритмів MD5 і SHA-1 до SHA-2 і SHA-3 то поява квантового комп'ютера спричинить необхідність багаторазового збільшення довжини ключів для симетричного шифрування, а для хеш-функцій – багаторазового збільшення довжини хеш. Такі довжини стають неприйнятними для практичного використання симетричних алгоритмів та існуючих хеш-функцій у майбутньому.

Вважається, що 2048-бітові ключі RSA забезпечуватимуть достатній рівень безпеки до 2030 року, а 3072 бітові ключі будуть беззастережно безпечними в найближчому майбутньому, проте, згідно з звітом NIST, у разі реалізації повномасштабного квантового комп'ютера, RSA та ECDSA стануть небезпечними. Деякі вчені дають тимчасову оцінку 20 років для реалізації такого повномасштабного пристрою.

Таким чином, у рамках проведеного дослідження [10] NIST поставило завдання щодо аналізу та синтезу існуючих пост-квантових підходів, а також аналіз кандидатів другого туру конкурсу стандартів NIST серед схем електронного підпису для виявлення найбільш перспективної.

Національний інститут стандартів і технологій вибирає один або кілька криптографічних алгоритмів з відкритим ключем через публічний процес, схожий на змагання. Нові стандарти криптографії з відкритим ключем визначать один або кілька додаткових цифрових підписів, шифрування з відкритим ключем і алгоритми встановлення ключа для доповнення Федерального стандарту обробки інформації (FIPS) 186-4, стандарту цифрового підпису (DSS), а також спеціальної публікації NIST (SP) 800-56A Версія 3, Рекомендація щодо парних схем встановлення ключа з використанням криптографії з дискретним логарифмом, і SP 800-56B Версія 2, Рекомендація щодо встановлення парних ключів із використанням криптографії з цілочисельною факторізацією. Передбачається, що ці алгоритми будуть здатні захищати конфіденційну інформацію в осяжному майбутньому, в тому числі після появи квантових комп’ютерів.

Процес стандартизації пост-квантової криптографії NIST розпочався у 2017 році з 69 алгоритмів-кандидатів, які відповідали як мінімальним критеріям прийняття, так і вимогам подання. Перший раунд тривав до січня 2019 року, під час якого алгоритми-кандидати були оцінені на основі їх безпеки, продуктивності та інших характеристик. NIST вибрав 26 алгоритмів для переходу до другого раунду для додаткового аналізу. Цей звіт описує

процес оцінки та відбору кандидатів у другому турі на основі відгуків громадськості та внутрішньої перевірки. У звіті підсумовуються 26 алгоритмів кандидатів у другий тур та визначаються ті, які відібрані для просування до третього раунду конкурсу. У фінал третього раунду увійшли алгоритми шифрування з відкритим ключем і встановлення ключа Classic McEliece, CRYSTALS-KYBER, NTRU та SABRE. Фіналістами третього раунду цифрових підписів стали CRYSTALS-DILITHIUM, FALCON та Rainbow. Ці фіналісти будуть розглянуті для стандартизації в кінці третього раунду. Крім того, вісім альтернативних алгоритмів-кандидатів також пройдуть до третього раунду: BIKE, FrodoKEM, HQC, NTRU Prime, SIKE, GeMSS, Picnic і SPHINCS+. Ці додаткові кандидати все ще розглядаються для стандартизації, хоча навряд чи це станеться наприкінці третього туру. NIST сподівається, що оголошення цих фіналістів і додаткових кандидатів приверне увагу криптографічної спільноти під час наступного раунду.

У звіті Національного Інституту Стандартів і Технологій США, NIST (квітень 2016 року)) зазначається, що більшість асиметричних криптографічних примітивів, що широко використовуються сьогодні в різних сферах суспільного життя, і які базуються на завданнях факторизації та дискретного логарифмування в різних групах, будуть скомпрометовані.

Критерії оцінки. У конкурсі пропозицій NIST було визначено три широкі аспекти критеріїв оцінки, які будуть використовуватися для порівняння алгоритмів-кандидатів у процесі стандартизації NIST PQC: 1) безпека, 2) вартість і продуктивність і 3) характеристики алгоритму та реалізації. Ці критерії описані нижче разом із обговоренням того, як вони вплинули на оцінку кандидатів у другому турі.

Безпека. Як і в минулих змаганнях Advanced Encryption Standard (AES) і Secure Hash Algorithm 3 (SHA-3), безпека є найважливішим фактором при оцінці кандидатів на постквантові алгоритми. Поточні стандарти відкритих ключів NIST використовуються в широкому спектрі додатків, включаючи

Інтернет-протоколи, такі як TLS, SSH, IKE, IPsec і DNSSEC, а також для сертифікатів, підпису програмного коду та безпечних завантажувачів. Потрібні нові стандарти для забезпечення безпеки для всіх цих програм. З метою кількісної оцінки безпеки алгоритмів-кандидатів NIST дав три можливі визначення безпеки — два для шифрування і одне для підписів. NIST також визначив п'ять категорій міцності безпеки для класифікації обчислювальної складності атак, які порушують визначення безпеки.

Для схем шифрування загального використання та встановлення ключів у конкурсі пропозицій [10] було запропоновано «семантично захищені» схеми щодо адаптивної атаки вибраного зашифрованого тексту (еквівалентно захист IND-CCA21). Для ефемерних випадків використання NIST також прийняв алгоритми, які забезпечують семантичну безпеку щодо вибраної атаки на відкритий текст (безпека IND-CPA), оскільки безпека IND-CCA2 не потрібна в суворо ефемерних випадках використання, і намагається задовольнити більш суворі вимоги IND- Безпека CCA2 може призвести до значного зниження продуктивності для деяких схем. Схеми цифрового підпису були необхідні для забезпечення екзистенційно непідробних підписів щодо адаптивної атаки на вибране повідомлення (безпека EUF-CMA). Відправників заохочували, але не вимагали надавати докази безпеки у відповідних моделях.

П’ять категорій міцності безпеки, визначені в [10], були засновані на обчислювальних ресурсах, необхідних для виконання певних атак грубої сили проти існуючих стандартів NIST для AES і SHA в різних моделях вартості обчислень, як класичних, так і квантових. Заявникам було запропоновано надати попередню класифікацію всіх запропонованих наборів параметрів відповідно до цих визначень. Хоча параметри категорій 1, 2 і 3 були (і залишаються) найважливішими цілями для оцінки NIST, NIST, тим не менш, наполегливо рекомендує авторам надати принаймні один набір параметрів, який відповідає категорії 5. Більшість алгоритмів-кандидатів уже зробили це; деякі не мають.

NIST також згадав інші бажані властивості безпеки, такі як ідеальна пряма конфіденційність, стійкість до атак з бічних каналів і кількох ключів, а також стійкість до неправильного використання, і все це продовжує представляти інтерес. Крім того, NIST вимагав пакетів подання, щоб узагальнити відомі криптоаналітичні атаки на схему та оцінки складності цих атак.

Під час першого та другого раундів процесу стандартизації NIST ряд результатів криптоаналітики різко знизили безпеку одних поданих схем і підірвали впевненість NIST у зрілості інших. Ці результати були основою для багатьох рішень NIST, які до цього часу були в процесі.

Оскільки NIST наближається до прийняття рішень щодо стандартизації, криптоаналіз, спрямований на точне вимірювання безпеки різних поданих наборів параметрів щодо відомих атак, стане ще більш актуальним. Це може включати, наприклад, точну кількісну оцінку частоти невдач дешифрування для наборів параметрів CCA, кількісну оцінку конкретної алгоритмічної складності решітки, а також розробку та тестування контрзаходів для розширених атак з боку каналу, таких як диференційний аналіз потужності. Дослідникам також було б корисно використовувати методи формальної перевірки, щоб забезпечити правильність доказів безпеки. Вся ця робота важлива для забезпечення високого ступеня впевненості, необхідного для NIST для стандартизації деяких із цих схем найближчим часом.

NIST також розглядає різноманітність припущень щодо складності обчислень, як важливу довгострокову мету безпеки для своїх стандартів. NIST сподівається стандартизувати практично ефективні схеми з різних сімейств криптосистем, щоб зменшити ризик того, що один прорив у криптоаналізі залишить світ без життєздатного стандарту для встановлення ключів або цифрових підписів. Тим не менш, NIST не відчуває потреби вибирати стандарти відразу, а скоріше віддасть пріоритет тим схемам, які готовимі до стандартизації та широкого впровадження. NIST вважає, що ця стратегія

найкраще підходить для збалансування бажання різноманітності з необхідністю ретельної перевірки всіх стандартів перед їх випуском.

Вартість та продуктивність. Оригінальний конкурс пропозицій [10] визначив вартість як другий найважливіший критерій при оцінці алгоритмів-кандидатів. Вартість включає обчислювальну ефективність генерації ключів і операцій з відкритими та приватними ключами, витрати на передачу відкритих ключів і підписів або шифротекстів, а також витрати на впровадження в термінах ОЗП (пам'яті з довільним доступом) або кількості воріт.

Під час другого раунду процесу стандартизації NIST PQC стала доступною більше інформації про обчислювальну ефективність алгоритмів. Для багатьох алгоритмів були передбачені швидші реалізації постійного часу на процесорах Intel x64, а також ARM Cortex-M4 та апаратні реалізації. Ці нові реалізації забезпечили кращу інформацію не тільки про продуктивність різних алгоритмів, але й про ресурси, необхідні для реалізації (оперативна пам’ять або кількість воріт). NIST сподівається побачити більше та кращі дані щодо ефективності в третьому раунді. Сподіваємося, ці дані про продуктивність включатимуть реалізації, які захищають від атак з боку каналу, таких як атаки на час, атаки моніторингу потужності, атаки збоїв тощо.

При порівнянні загальної продуктивності алгоритмів враховувалися як обчислювальні витрати, так і вартість передачі даних. З цією метою алгоритми, як правило, порівнювалися з двома окремими категоріями, залишаючи відкритою можливість того, що різні алгоритми в кінцевому підсумку будуть стандартизовані для цих двох категорій випадків використання.

Для загального використання при оцінці загальної продуктивності враховувалися витрати на передачу відкритого ключа на додаток до підпису або зашифрованого тексту під час кожної транзакції. Для KEM перевага надавалася алгоритмам, які забезпечували кращу загальну продуктивність з

урахуванням вартості генерації ключа, оскільки багато програм використовують нову пару ключів KEM для кожної транзакції, щоб забезпечити пряму секретність. Для алгоритмів підпису вартість генерації ключа вважалася менш важливою.

Для використання спеціального призначення вимоги до продуктивності можуть дещо відрізнятися, і краще використовувати різні алгоритми. Наприклад, деякі з алгоритмів-кандидатів є обчислювально ефективними і мають дуже маленькі підписи або зашифровані тексти, але дуже великі відкриті ключі. Хоча ці алгоритми можуть не забезпечити найкращу загальну продуктивність, коли відкритий ключ потрібно надіслати для кожного з'єднання, існують програми, для яких відкритий ключ розповсюджується заздалегідь (наприклад, як частина програмного пакету), так що вартість передачі відкритий ключ не входить у вартість транзакції. Алгоритми з великими відкритими ключами можуть добре працювати з цими програмами, навіть якщо вони не забезпечують найкращу продуктивність для інших програм.

Алгоритм і характеристики реалізації. Хоча поточні реалізації алгоритмів-кандидатів мають тенденцію працювати в постійному часі, можуть виникнути тонкі проблеми, які можна не помітити (див., наприклад, [11]). Більшість реалізацій не забезпечують захист від інших типів атак на бічні канали, таких як аналіз потужності. Під час третього раунду NIST сподівається зібрати більше інформації про витрати на впровадження цих алгоритмів таким чином, щоб забезпечити стійкість до таких атак. Реалізації на дуже обмежених пристроях, таких як смарт-карти, як правило, більш вразливі до таких атак, ніж реалізації на комп'ютерах загального призначення, оскільки пристрої з обмеженими можливостями, швидше за все, будуть використовуватися в середовищах, у яких зловмисник має необмежений доступ до пристрою. Таким чином, окрім вартості обчислювального часу, вартість реалізації цих

пом’якшення з точки зору оперативної пам’яті (або кількості воріт) також дуже важлива.

Крім того, NIST вивчив потенційний вплив на продуктивність алгоритмів-кандидатів у існуючих широко використовуваних протоколах (наприклад, TLS, IPSec і SSH) і сертифікатах. Зрозуміло, що деякі алгоритми можуть спричинити серйозні проблеми з продуктивністю, якщо їх включити в ці протоколи — особливо схеми з дуже великими відкритими ключами, зашифрованими текстами або підписами. Використання цих алгоритмів у таких широко використовуваних протоколах вимагало б значної реорганізації для досягнення належної продуктивності. Інші алгоритми-кандидати можна замінити на існуючі алгоритми створення підписів і ключів з відносно невеликою втратою продуктивності.

Декілька кандидатів внесли оновлення під час першого та другого турів, щоб покращити простоту. Розглядаючи питання стандартизації, переважні прості та елегантні конструкції, оскільки вони спонукають до подальшого аналізу та розуміння. NIST і надалі приділятиме пильну увагу тому, наскільки добре проаналізовані та добре зрозумілі кандидати під час третього туру.

Як зазначено в [10], NIST розглядатиме будь-які фактори, які можуть перешкодити або сприяти прийняттю алгоритму чи реалізації, включаючи, але не обмежуючись, інтелектуальну власність, що охоплює алгоритм чи його реалізації, а також наявність та умови ліцензій для зацікавлених сторін. У третьому раунді міркування, пов’язані з інтелектуальною власністю, можуть зіграти більш важливу роль, оскільки NIST приймає рішення щодо стандартизації. NIST віддає перевагу безоплатним алгоритмам, щоб забезпечити широке застосування.

2.2 Алгоритм NTRUEncrypt і NTRUSign

Алгоритми NTRUEncrypt і NTRUSign є на даний момент найбільш ефективними втіленнями алгоритмів з відкритим ключем, безпека яких ґрунтується на складностsі задачи пошуку коротокого вектору решітки.

Решітки вивчені криптографами протягом деякого часу, як у сфері криптоаналізу, так і у джерелі важких проблем, на яких створює схеми шифрування. Ранні схеми були обумовлені Голдрейхом, Голдвассер та Халевим, які запропонували, щоб схема підписавала повідомлення, демонструючи здатність вирішувати приблизну найближчу векторну проблему.

NTRUEncypt, схема була запропонована приблизно в той же час, як і NTRUSign. Як Один з декількох способів, що не можна переглянути, - це криптосистема решітки, заснована на особливо ефективному класі модульних решіток, які будуть посилатися на решітки NTRU. Приватний ключ для схеми шифрування NTRUEncypt складається з хорошої основи для V-мірної підпорядкування 2V-мірної решітки NTRU, але для того, щоб ефективно вирішувати оцінку CVP для довільного дайджесту повідомлень, потрібно знати повну гарну основу для решіток.

Алгоритм NTRUEncrypt зазвичай описується як поліноміальна криптосистема, що включає продукти згортки. Природно, що її також можна розглядати як решіткову криптосистему для певного обмеженого класу решіток.

Криптосистема має кілька природних параметрів, як і у всіх практичних криптосистемах. Один з найцікавіших методів криптоаналітики на сьогодні, що стосується NTRUEncrypt, використовує властивість, що за певних параметрів криптосистема не може належним чином розшифрувати дійсні шифротексти. Функціональність криптосистеми не зазнає негативного впливу,

коли ці, так звані, «збої дешифрування» відбуваються з дуже малою ймовірністю у випадкових повідомленнях, але зловмисник може вибрати повідомлення, щоб викликати збій, і припускаючи, що він знає, коли повідомлення не вдалося виконати дешифрування (що є типовою моделлю безпеки в криптографії) існують ефективні способи вилучення приватного ключа зі знання невдалих зашифрованих текстів (тобто помилки дешифрування сильно залежать від ключа). Це було вперше помічено і є важливим фактором при виборі параметрів для NTRUEncrypt.

Інші міркування щодо безпеки для параметрів NTRUEncrypt включають оцінку безпеки криптосистеми від скорочення решітки, атак зустріч посередині на основі структури приватного ключа NTRU та гібридних атак.

Алгоритм NTRUSign. Пошук схеми підпису на основі «нульового знання» є відкритою проблемою в криптографії. Варто зауважити, що більшість криптографів припускають, що будь-яка схема підпису автоматично матиме властивість «нульового знання», тобто визначення схеми підпису передбачає, що проблеми визначення приватного ключа або створення підробок повинні стати не легше після того, як ви побачите поліноміальну кількість дійсних підписів. Проте в теорії решіток схеми сигнатур з аргументами редукції тільки з'являються, і їх обчислювальна ефективність зараз досліджується. Для більшості схем підпису на основі решітки відомі явні атаки, які використовують знання, отримані з розшифровки підписів.

Перші спроби створення таких практичних схем підпису на основі NTRU піддалися атакам. Відомі атаки на NTRUSign, рекомендовану на даний момент схему підпису, вимагають довжини стенограми непрактичної довжини, тобто схема підписів на даний момент має практичне значення.

NTRUSign був винайдений між 2001 і 2003 роками винахідниками NTRUEncrypt разом з Н. Хоугрейв-Гремом і В. Уайтом. Як і NTRUEncrypt, він дуже параметризується і зокрема, має параметр, що включає кількість збурень.

Найцікавіший криптоаналітичний прогрес у NTRUSign показав, що його необхідно використовувати принаймні з одним збуренням, тобто існує ефективна та елегантна атака, яка вимагає невеликого запису підписів у випадку нульових збурень.

NTRUEncrypt параметри.

Реалізація примітиву шифрування NTRUEncrypt визначається такими параметрами:

N Параметр ступеня. Додатне ціле число. Асоційована решітка NTRU має розмірність 2N.

q Великий модуль. Додатне ціле число. Асоційована решітка NTRU є модульною решіткою згортки модуля q.

p Малий модуль. Ціле число або поліном.

D_f, D_g Приватні ключі. Набори малих поліномів, з яких вибираються закриті ключі.

D_m Простір відкритого тексту. Набір поліномів, які представляють шифровані повідомлення. Обов'язком схеми шифрування є забезпечення методу кодування повідомлення, яке потрібно зашифрувати, у поліном у цьому просторі.

D_r Сліпучий простір цінностей. Набір поліномів, з яких вибирається тимчасове сліпуче значення, що використовується під час шифрування.

center Метод центрування. Засіб виконання зменшення мод q при дешифруванні.

Генерація ключа NTRUEncrypt складається з таких операцій:

1) Випадкове генерування поліномів f і g у D_f, D_g відповідно.
2) Інвертуйте f у R_q, щоб отримати f_q, інвертувати f у R_p, щоб отримати f_p, і перевірте що g є оборотним у R_q.

3) Відкритий ключ h = p * g * f_q (mod q). Закритим ключем є пара (f, f_p).

Шифрування NTRUEncrypt складається з таких операцій:

1) Довільно виберіть «малий» поліном $r \in D_r$.
2) Обчисліть зашифрований текст e як $e \equiv r * h + m\ (mod\ q)$.

Розшифрування NTRUEncrypt складається з таких операцій:

1) Розрахувати $a \equiv center(f * e)$, де операційний центр центрування зменшує свій вхід в інтервал.
2) Відновіть m шляхом обчислення $m \equiv f_p * a\ (mod\ p)$.

Щоб зрозуміти, чому розшифровка коректна, використовується формула: $h \equiv p * g * f_q$ і $e \equiv r * h + m$.

$$a \equiv p * r * g + f * m\ (mod\ q).$$

Для відповідного вибору параметрів і center це рівність над Z, а не просто над Z_q. Отже, крок 2 відновлює m: термін p * r * g зникає, і f_p * f * m = m (mod p).

Реалізація примітиву NTRUSign використовує такі параметри:

N Поліноми мають ступінь < N

q Коефіцієнти поліномів зменшуються за модулем q

D_f, D_g Поліноми в $\mathcal{T}\ (d)$ мають коефіцієнти d + 1 що дорівнюють 1, коефіцієнти d є рівні -1, а інші коефіцієнти дорівнюють 0.

$\mathcal{N}$ Прив'язка до норми, що використовується для перевірки підпису.

β Коефіцієнт балансу для норми $|\cdot|_\beta$. Має властивість $0 < \beta \leq 1$.

Генерація ключа NTRUSign складається з таких операцій:

1) Генерувати випадковим чином «малі» поліноми f і g у D_f, D_g , такі, що f і g є оберненими за модулем q.

2) Обчислюються поліноми F і G такі, що
$$F * G - g * F = q,$$
I F і G мають розмір
$$\| F \| \approx \| G \| \approx \| f \| \sqrt{N/12}.$$
Методи обчислення наведено в [17].
3) Позначимо обернену до f в R_q через f_q, а обернену до g в R_q через g_q. Відкритий ключ $h = F * f_q \pmod q = G * g_q \pmod q$. Закритим ключем є пара (f, g).

Операція підписання передбачає округлення поліномів. Для будь-якого $a \in Q$, нехай [a] позначає ціле число, найближче до a, і визначте, $\{a\} = a - [a]$. (Для чисел a, які знаходяться посередині між двома цілими числами, ми вказуємо, що $\{a\} = +\frac{1}{2}$, а не $-\frac{1}{2}$.) Якщо A - поліном з раціональними (або дійсними) коефіцієнтами, нехай [A] і {A} - A з зазначеною операцією, застосованою до кожного коефіцієнта.

Перевірка включає використання балансувального коефіцієнта β та норми, що визначається $\mathcal{N}$. Щоб перевірити, одержувач робить наступне:

1) Відобразити цифровий документ D, що підлягає верифікації, у вектор $m \in [0, q]^N$ за допомогою узгодженої хеш-функції.
2) Обчисліть t = s * h mod q, де s — підпис, а h — відкритий ключ підписувача.
3) Обчисліть норму
$$v = \min_{k_1, k_2 \in R} (||s + k_{1q}||^2 + \beta^2 ||(t - m) + k_{2q}||^2)^{1/2}$$
4) Якщо $v \leq \mathcal{N}$, перевірка проходить успішно, інакше не вдається.

Чому NTRUSign працює. Для будь-яких натуральних чисел N і q і будь-якого полінома $h \in R$ ми можемо побудувати решітку L_h, що міститься в $R^2 \cong Z^{2N}$, таким чином:
$$L_h = L_h(N, q) = \{(r, r') \in R \times R \mid r' \equiv r * h \ (mod\ q)\}$$

Цю підгратку Z^{2N} називають модульною ґраткою згортки. Він має розмірність $2N$ і визначник q^N.

Оскільки

$$\det\begin{pmatrix} f & F \\ g & G \end{pmatrix} = q$$

і ми визначили h = F/f = G/g mod q, знаємо що

$$\begin{pmatrix} f & F \\ g & G \end{pmatrix} \text{ і } \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}$$

є підставами для тієї ж решітки. Тут, як і в цьому випадку, матриця поліномів 2 на 2 перетворюється на цілу матрицю 2N на 2N шляхом перетворення кожного полінома в поліноміальній матриці до його представлення у вигляді циркуляційної матриці N на N, і два уявлення розглядаються як еквівалентні.

Підписування полягає у знаходженні близької точки решітки до точки повідомлення (0, m) за допомогою методу Бабая: виразіть цільову точку як комбінацію базисних векторів із дійсним значенням і знайдіть близьку точку решітки, округляючи дробові частини дійсних коефіцієнтів, щоб отримати цілі комбінації базисних векторів. Помилка, внесена цим процесом, буде сумою помилок округлення для кожного з базових векторів, а помилка округлення за визначенням буде від $-\frac{1}{2}$ до $\frac{1}{2}$. У NTRUSign всі базові вектори мають однакову довжину, тому очікувана похибка, яку вносять $2N$ округлень цього типу, буде у $\sqrt{N/6}$ разів на цю довжину.

У NTRUSign приватний базис вибирається таким чином, що ||f|| = ||g|| і $||F|| \sim ||G|| \sim \sqrt{N/12}||f||$. Отже, очікувана помилка при підписанні буде

$$\sqrt{N/6}||f|| + \beta(\frac{N}{6\sqrt{2}})||f||.$$

Навпаки, зловмисник, який використовує лише відкритий ключ, швидше за все, створить підпис з N неправильними коефіцієнтами, і ці коефіцієнти будуть розподілені випадковим чином по mod q. Отже, очікувана помилка при генерації підпису з відкритим ключем

$$\beta\sqrt{N/12}q$$

(Більш детально в розділі 3.2 і далі; мета цього розділу полягає в тому, щоб стверджувати, що закритий ключ дозволяє створювати менші підписи, ніж відкритий ключ).

Тому зрозуміло, що можна вибрати ||f|| і q таким чином, щоб знання приватної бази дозволяло створювати менші помилки підписання, ніж знання лише публічної бази. Таким чином, гарантуючи, що помилка підписання менша, ніж можна було б очікувати на публічній базі, одержувач може перевірити, що підпис був створений власником приватної бази і, отже, є дійсним.

Інженерна специфікація NTRUSign. Для інженерних цілей NTRUSign визначається з точки зору операцій з набору R поліномів ступеня (строго) менше ніж N і мають цілі коефіцієнти, параметр N фіксований. Основними операціями над цими поліномами є додавання та множення згортки. Множення згортки двох поліномів f і g визначається шляхом прийняття коефіцієнта X^k у f * g.

$$(f * g)_k \equiv \sum_{i+g \equiv k\ (mod\ N)} f_i * g_j \quad (0 \leq k < N).$$

У математичних термінах R – це кільце зрізаних поліномів R = Z [X] / (X^N–1). Якщо один із поліномів має всі коефіцієнти, вибрані з набору {0,1} то така згортка називається двійковою. Якщо коефіцієнти поліномів зменшуються по модулю q для деяких q, ми будемо посилатися на згортку як модульну.

Також потрібно буде округлити числа до найближчого цілого і взяти їх дробові частини. Для будь-якого a Є Q нехай [a] позначає ціле число, найближче до a, і визначимо {a} = a - [a]. Якщо A є поліномом з раціональними (або дійсними) коефіцієнтами, нехай [A] і {A} буде A із зазначеною операцією, застосованою до кожного коефіцієнта.

2.3 Переможці 2 туру конкурсу NICT

NIST відібрав 26 алгоритмів-кандидатів до другого туру. З 26 кандидатів 17 були механізмами встановлення ключа (KEM) або схемами шифрування з відкритим ключем, а дев’ять — цифровими підписами. Командам, які подали документи, було дозволено вносити незначні зміни та повторно подавати свої пакети, які мали відповідати тим самим вимогам, що й оригінальні подані документи. Чотири з кандидатів були об'єднані алгоритмами першого раунду: LEDAcrypt (об'єднаний з LEDAkem і LEDApkc), NTRU (об'єднаний з NTRUEncrypt і NTRU-HRSS-KEM), ROLLO (об'єднаний з LAKE, LOCKER і Ouroboros-R) і Round5 (об’єднано з Hila5 та Round2).

Таблиця 2.1 Кандидати у другий тур.

BIKE	LEDAcrypt	Rainbow
Classic McEliece	LUOV	ROLLO
CRYSTALS-DILITHIUM	MQDSS	Round5
CRYSTALS-KYBER	NewHope	RQC
FALCON	NTRU	SABER
FrodoKEM	NTRU Prime	SIKE
GeMSS	NTS-KEM	SPHINCS+
HQC	Picnic	Three Bears
LAC	qTESLA	

Отже, кандидати на новий постквантовий стандарт цифрового підпису:

CRYSTALS-DILITHIUM [2] — Є представником криптографії на решітках. За основу взято схему Фіата-Шаміра [1] з перериваннями.

Криптоаналіз зводиться до розв'язання задач Module-LWE та Module-SIS. Має хорошу продуктивність та може бути ефективно реалізована на малоресурсних пристроях. NIST висунув вимогу додати набір загальносистемних параметрів для 5-го рівня безпеки.

FALCON [4] — Також є представником криптографії на решітках. Але за основу взято фреймворк GPV [4]. Криптоаналіз зводиться до завдання SIS на решітках NTRU. Головним недоліком цієї схеми є складна програмна та апаратна реалізація. Схема використовують обчислення над числами з плаваючою комою, що дуже ускладнює як аналіз стійкості до атак по стороннім каналам, і робить складним реалізацію малоресурсних пристроїв.

Rainbow [6] — є основою криптографії на мультиваріативних перетвореннях. За основу взято схему UOV [5]. Головною перевагою є розмір цифрового підпису. Але з-за великого розміру ключа цю схему рекомендується використовувати лише специфічних завдань, де розмір ключів не критичний.

NIST також заявили, що хоча б одна зі схем CRYSTALS-DILITHIUM [10] і FALCON буде стандартизована. Таким чином, для цифрового підпису, швидше за все, в майбутньому будуть використовуватися схеми на основі криптографії на решітках. А для більш специфічних завдань – Rainbow.

Для асиметричного шифрування в третій етап вийшли:

Classic McEliece [6] — Є представником криптографії на кодах, що виправляють помилки. Основна конструкція схеми була запропонована ще 1979 року і добре вивчена. Має малі розміри шифротекстів, але дуже великий розмір ключа. Через це має ті ж проблеми, що й Rainbow і рекомендується використовувати лише для специфічних завдань.

CRYSTALS-KYBER [7] — Є представником криптографії на ґратах. Криптоаналіз зводиться до розв'язання задачі Module-LWE. Для забезпечення стійкості до атак із адаптивно підібраними шифротекстами використовується

перетворення Фуджісакі-Окамото. Має хорошу продуктивність і безпеку, але NIST також нагадують, що Module-LWE — це відносно маловивчена проблема і потребує більш детального криптоаналізу.

NTRU – [3] Є представником криптографії на ґратах. За основу взято схему NTRUEncryt, запропоновану понад 20 років тому. Проблема NTRU, на відміну Module-LWE (та інших модифікацій) була дуже добре вивчена, що дуже важливим чинником.

SABER – [6] Є представником криптографії на ґратах. Криптоаналіз зводиться до проблеми MLWR (Module-LWE, де замість складання з вектором помилки використовується заокруглення за меншим модулем). Використовується перетворення Фуджісакі-Окамото, як і в CRYSTALS-KYBER.

Загалом ситуація аналогічна — для загального використання рекомендуються схеми на основі ґрат. Але NIST зауважили, що тільки одна зі схем на ґратах (CRYSTALS-KYBER, NTRU, SABER) буде стандартизована.

Альтернативні схеми. Також NIST відібрали 8 альтернативних схем, які не увійшли до фіналу, але є перспективними.

2.3.1 Алгоритм CRYSTALS-Kyber та його специфікації

Kyber — це IND-CCA2-захищений механізм інкапсуляції ключів (KEM), який вперше був описаний в. Безпека Kyber заснована на жорсткості вирішення проблеми навчання з помилками в решітках модулів (проблема MLWE). Побудова Kyber здійснюється двоетапним підходом: спочатку вводиться захищена INDCPA схема шифрування з відкритим ключем, яка шифрує повідомлення фіксованої довжини 32 байти, яка називається

Kyber.CPAPKE. Потім використовується дещо змінене перетворення Фудзісакі-Окамото (FO), щоб побудувати захищений IND-CCA2 KEM.

Байти та байтові масиви. Вхідні та вихідні дані для всіх функцій API Kyber є масивами байтів. Для спрощення записів позначимо через β множину {0, ……., 255}, тобто набір 8-розрядних цілих чисел без знака (байтів). Отже, через B^k позначимо множину байтових масивів довжини *k*, а через B^* - множину байтових масивів довільної довжини (або потоків байтів). Для двох байтових масивів *a* і *b* позначаємо (a||b) конкатенацію *a* і *b*. Для байтового масиву *a* позначаємо *a* + *k* масив байтів, що починається з байта *k* до байта *a* (з індексацією, починаючи з нуля). Наприклад, нехай *a* на байтовий масив довжини *l*, нехай *b* буде іншим масивом байтів і нехай c = (a||b) буде конкатенацією *a* і *b*; тоді *b* = *a* + *l*. Коли зручніше працювати з масивом бітів, ніж з масивом байтів, робимо це перетворення явним за допомогою функції BytesToBits, яка приймає як вхідний масив *l* байтів і виробляє як вихідний масив з *8l* біт. Біт β_i в позиції *i* вихідного бітового масиву отримується з байта $b_{i/8}$ в позиції i/8 вхідного масиву шляхом обчислення $\beta_i = ((b_{i/8}/ 2^{(i \bmod 8)}) \bmod 8)$.

Поліноміальні кільця та вектори. Позначимо через *R* кільце $Z[X]/(X^n+1)$, а через R_q — кільце $Z_q[X]/(X^n+1)$, де $n = 2^{n'-1}$ таке, що $X^n + 1$ є $2^{n'}$-й цикломічний поліном. У цьому документі значення *n, n'* і q фіксуються на *n* = 256, *n'* = 9 і *q* = 3329. Звичайні літери шрифту позначають елементи в *R* або R_q (який включає елементи в Z і Z_q) і жирний нижній - літери регістру представляють вектори з коефіцієнтами в *R* або R_q. За замовчуванням усі вектори будуть векторами-стовпцями. Жирні великі літери є матрицями. Для вектора *v* (або матриці **A**) позначимо через $\boldsymbol{v}^T$ (або $\boldsymbol{A}^T$) його транспонування. Для вектора *v* пишемо *v*[*i*], щоб позначити його *i*-й запис (з індексацією, починаючи з нуля); для матриці $\boldsymbol{A}$ пишемо $\boldsymbol{A}[i][j]$, щоб позначити запис у рядку *i*, стовпці *j* (знову ж таки, з індексацією, починаючи з нуля).

Модульні скорочення. Для парного (відповідно непарного) натурального числа α визначаємо r’ = r $\mathrm{mod}^{\pm}$ α як єдиний елемент r’ в діапазоні $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (відповідно $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) такий, що r’ = r mod α. Для будь-якого натурального числа α визначаємо r’ = r mod^{+}α як єдиний елемент r’ в діапазоні $0 \leq r' < \alpha$, такий, що r’ = r mod α. Коли точне представлення не важливе, пишемо r mod α.

Округлення. Для елемента $x \in Q$ позначимо $[x]$ округлення x до найближчого цілого числа із округленням зв’язків у більшу сторону.

Розміри елементів. Для елемента $w \in Z_q$ пишемо $||w||_\infty$ як $|w \ mod^{\pm} \ q|$. Тепер визначимо норми l_∞ і l_2 для $w = w_0 + w_1X + \ldots + w_{n-1}X^{n-1} \in R$:

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

Аналогічно, для $w = (w_1, \ldots, w_k) \in R_k$ визначимо

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

Набори та розподіли. Для набору S пишуть $s \leftarrow S$, щоб позначити, що s вибирається випадковим чином із S. Якщо S є розподілом ймовірностей, то це означає, що s вибирається відповідно до розподілу S.

Рівномірна вибірка в R_q. Kyber використовує детермінований підхід до елементів вибірки в R_q, які статистично близькі до рівномірно випадкового розподілу. Для цієї вибірки використовуємо функцію Parse: $B^* \rightarrow R_q$, яка отримує на вхід потік байтів $B = b_0, b_1, b_2, \ldots$ і обчислює NTT-подання $\hat{a} = \hat{a}_0 + \hat{a}_1X + \cdots + \hat{a}_{n-1}X^{n-1} \in R_q$ для $a \in R_q$.

Інтуїція за функцією Parse полягає в тому, що якщо вхідний масив байтів статистично близький до рівномірно випадкового масиву байтів, то вихідний поліном статистично близький до рівномірно випадкового елемента R_q. Він представляє рівномірно випадковий поліном в R_q, оскільки NTT є біективним

і, таким чином, відображає поліноми з рівномірно випадковими коефіцієнтами в поліноми з знову рівномірно випадковими коефіцієнтами.

Вибірка з біноміального розподілу. Шум у Kyber відбирається з центрованого біноміального розподілу B_η для $\eta = 2$ або $\eta = 3$. Визначаємо B_η таким чином:

$$\text{Sample } (a_1, \dots, a_\eta, b_1, \dots, b_\eta) \leftarrow \{0,1\}^{2\eta}$$

$$\text{and output } \sum_{i=1}^{\eta}(a_i - b_i)$$

Коли пишемо, що поліном $f \in R_q$ або вектор таких поліномів відбирається з B_η, мається на увазі, що кожен коефіцієнт відбирається з B_η.

Для специфікації Kyber нам потрібно визначити, як поліном $f \in R_q$ вибирається відповідно до B_η детерміновано з 64η байтів виводу псевдовипадкової функції (фіксуємо $n = 256$ в цьому описі). Це робиться за допомогою функції CBD (для «центрованого біноміального розподілу»).

Кодування та декодування. Існує два типи даних, які Kyber потрібно серіалізувати до байтових масивів: байтові масиви та (вектори) поліномів. Байтові масиви тривіально серіалізуються через ідентичність, тому нам потрібно визначити, як серіалізуємо та десеріалізуємо поліноми. В Алгоритмі 3 дається псевдокодовий опис функції Decode', яка десеріалізує масив із $32l$ байтів у поліном $f = f_0 + f_1X + \dots + f_{255}X^{255}$ (знову фіксується $n = 256$ в цьому описі) з кожним коефіцієнтом f_i у $\{0, \dots, 2^l\text{-}1\}$. Визначається функцію Encode_l як зворотну до Decode_l. Щоразу, коли застосовуємо Encode_l до вектора поліномів, кодуємо кожен поліном окремо та об’єднуємо вихідні масиви байтів.

Специфікація Kyber.CPAPKE [13] схожа на схему шифрування LPR, яку представили (для Ring-LWE) Любашевський, Пейкерт і Регев у презентації на Eurocrypt 2010 [8]; опис також є в повній версії статті. Коріння цієї схеми сягають першої схеми шифрування на основі LWE, представленої Регевом у,

з основною відмінністю в тому, що базове кільце не є Z_q, і як секретний, так і вектор помилок мають малі коефіцієнти. Ідея використання поліноміального кільця (замість Z_q) сходить до криптосистеми NTRU, представленої Хоффстайном, Пайфером та Сільверманом у [12], тоді як симетрія між секретом і помилкою вже використовувалася в дуже схожих криптографічних схемах.

Основна відмінність від схеми шифрування LPR полягає у використанні Module-LWE замість Ring-LWE. Крім того, використовується підхід, використаний Алкімом, Дукасом, Пеппельманом і Швабе для генерації загальнодоступної матриці A [14]. Крім того, треба скоротити шифротексти, заокруглюючи нижні біти, як у схемах, що базуються на навчанні [15], що є поширеною технікою для зменшення розміру зашифрованого тексту також у схемах на основі LWE.

Параметри. Kyber.CPAPKE параметризовано цілими числами n, k, q, η_1, η_2, d_u і d_v. Як зазначалося раніше, у цьому документі n завжди дорівнює 256, а q завжди дорівнює 3329.

Специфікація Kyber.CCAKEM. Було створено Kyber.CCAKEM IND-CCA2-захищеного KEM з IND-CPA-захищеного схемами шифрування з відкритим ключем, описаної в попередньому підрозділі, за допомогою дещо зміненого Fujisaki-Okamoto. В алгоритмах 1, 2 визначається генерацію ключів, інкапсуляцію та декапсуляцію Kyber.CCAKEM.

Алгоритм 1 Kyber.CCAKEM.KeyGen()

Output: Public Key $pk \in B^{\frac{12*k*n}{8+32}}$

Output: Secret Key $sk \in B^{\frac{24*k*n}{8+96}}$

1) $z \leftarrow B^{32}$
2) $(pk, sk') :=$ Kyber.CCAKEM.KeyGen()
3) $sk := (sk' \| pk \| \mathrm{H}(pk) \| z)$

4) return (pk, sk)

Алгоритм 2 Kyber.CCAKEM.Enc(pk, m, r)

Input**:** Public key $pk \in B^{\frac{12*k*n}{8+32}}$

Output**:** Ciphertext $c \in B^{du * k * n/8+du * n/8}$

Output**:** Shared key $K \in B^*$

1. $m \leftarrow B^{32}$
2. $m \leftarrow H(m)$
3. $(\overline{K}, r) \coloneqq G(m||\mathrm{H}(pk)$
4. $c \coloneqq$ Kyber.CPAPKE.Enc(pk, m, r)
5. $K \coloneqq$ KDF $(\overline{K}||\mathrm{H}(c))$
6. return (c, K)

Набори параметрів Kyber. Визначається три набори параметрів для Kyber, які називаються Kyber512, Kyber768 і Kyber1024. Параметри наведено в таблиці 1. Зверніть увагу, що в таблиці також наведено похідний параметр δ, який є ймовірністю того, що декапсуляція дійсного зашифрованого тексту Kyber.CCAKEM не вдасться. Параметри були отримані за допомогою наступного підходу:

Таблиця 2.2 Набори параметрів для Kyber.

	n	k	q	η_1	η_2	(d_u, d_v)	δ
Kyber512	256	2	3329	3	2	(10, 4)	2^{-139}
Kyber768	256	3	3329	2	2	(10, 4)	2^{-164}
Kyber1024	256	4	3329	2	2	(11, 5)	2^{-174}

– n встановлено на 256, оскільки мета полягає в тому, щоб інкапсулювати ключі з 256 бітами ентропії (тобто використовувати розмір відкритого тексту 256 біт у Kyber.CPAPKE.Enc). Менші значення n вимагають кодування кількох ключових бітів в один поліноміальний коефіцієнт, що вимагає нижчого рівня шуму і, отже, знижує безпеку. Більші значення n

зменшать можливість легкого масштабування безпеки за допомогою параметра k.

- Вибираємо q як мале просте просте число, що задовольняє $n \mid (q - 1)$; це потрібно, щоб увімкнути швидке множення на основі NTT. Є два менших простих числа, для яких ця властивість виконується, а саме 257 і 769. Однак для цих простих чисел ми не зможемо досягти нікчемної ймовірності відмови, необхідної для безпеки CCA, тому ми обрали наступне за величиною, тобто $q = 3329$.
- k вибирається для фіксації розміру решітки як кратного n; змінна k є основним механізмом у Kyber для масштабування безпеки (і, як наслідок, ефективності) до різних рівнів.
- Решта параметри η_1, η_2, d_u та d_v були обрані для балансу між безпекою, розміром шифрованого тексту та ймовірністю збою. Зауважте, що всі три набори параметрів досягають ймовірності відмови $< 2^{-128}$ з деяким запасом.
- Параметр η_1 визначає шум s, e, r в алгоритмах. Параметр η_2 визначає шум e_1 і e_2 в Алгоритмах.

Обґрунтування конструкції. Конструкція Kyber базується на версії модуля схеми шифрування Ring-LWE LPR з відкиданням бітів. Це також посилюється багатьма вдосконаленнями попередніх реалізацій схем шифрування на основі решітки, таких як NewHope. У NewHope (і в усіх інших схемах Ring-LWE) операції мали вигляд As + e, де всі змінні були поліномами в деякому кільці. Основна відмінність у Kyber полягає в тому, що A тепер є матрицею (з невеликою розмірністю, наприклад 3) над кільцем поліномів постійного розміру та s; e — вектори над одним кільцем. Ця схема називається «модульними решітками».

Використання Module-LWE. Попередні пропозиції криптосистем на основі LWE використовували або дуже структуровану проблему Ring-LWE (як, наприклад, NewHope), або стандартний LWE (як, наприклад, Frodo). Основною перевагою структурованих варіантів LWE на основі

поліноміальних кілець є ефективність з точки зору як швидкості, так і розмірів ключа та зашифрованого тексту. Недоліками є занепокоєння, що додаткова структура може забезпечити більш ефективні атаки і що компроміс між ефективністю та безпекою можна масштабувати лише досить грубо. Перевагами стандартного LWE є відсутність структури та легка масштабованість, але це відбувається за ціною значного зниження ефективності. Module-LWE пропонує компроміс між цими двома крайнощами. У конкретному випадку параметрів Module-LWE, які використовуються в Kyber, отримуємо дещо зменшену структуру в порівнянні з Ring-LWE, набагато кращу масштабованість і—при шифруванні повідомлень фіксованого розміру 256 біт продуктивність дуже схожа на продуктивність на основі Ring-LWE схеми.

Активна охорона. Бос, Костелло, Неріг і Стебіла використовували пасивно захищений KEM для переходу TLS до перехідної постквантової безпеки (тобто постквантової конфіденційності, але лише до квантової аутентифікації). Подальші роботи, такі як NewHope або Frodo, були продовжені та запропонували більш ефективні та консервативніші варіанти базового пасивно безпечного KEM. Однією з переваг пасивно захищених KEM є те, що вони можуть приймати більшу ймовірність відмови (що дозволяє підвищити безпеку за рахунок збільшення шуму або зменшення розміру відкритого ключа і шифротексту). Інша перевага полягає в тому, що вони не вимагають перетворення CCA, а отже, мають швидшу декапсуляцію. Незважаючи на ці переваги, Kyber визначається лише як захищений KEM IND-CCA2. Для багатьох програм, таких як шифрування з відкритим ключем (через конструкцію KEM-DEM) або при обміні автентифікованими ключами активна безпека є обов’язковою. Однак у випадках використання (наприклад, обмін ключами в TLS), які, строго кажучи, не вимагають активної безпеки, використання активно безпечного KEM має переваги. Зокрема, він дозволяє (навмисно чи випадково) кешувати ефемерні ключі. Крім того, CCA-перетворення Kyber захищає від певних помилок у реалізації. Зокрема,

пасивно захищені схеми не помітять, якщо партнер по комунікації використовує «неправильний» шум, наприклад, шум без нуля. Така помилка в інкапсуляції Kyber буде негайно уловлена кроком повторного шифрування під час декапсуляції. Як висновок, можна вважати, що накладні витрати на забезпечення безпеки CCA недостатньо великі, щоб виправдати його збереження та зробити схему менш надійною.

Роль НТТ. Множення в R_q на основі теоретичного перетворення чисел (NTT) має ряд переваг: воно надзвичайно швидке, не вимагає додаткової пам'яті (як, наприклад, множення Карацуби або Toom) і може виконуватися за дуже мало кодового простору. Таким чином, стало звичайною практикою вибирати параметри криптографії на основі решітки для підтримки цього дуже швидкого алгоритму множення. Деякі схеми йдуть далі і роблять NTT частиною визначення схеми. Яскравим прикладом знову є NewHope, який вибирає загальнодоступне значення a в домені NTT, а також надсилає відкриті ключі та зашифровані тексти в домені NTT, щоб зберегти 2 NTTs. NewHope була не першою схемою, яка робила це.

У Kyber також вирішили зробити NTT частиною визначення схеми, але лише у вибірці A та відкритого ключа, а не для формату зашифрованого тексту. Наслідком цього рішення є те, що NTT з'являється в специфікації Kyber:CPAPKE. Зауважте, що множення на A мають використовувати NTT просто тому, що $\hat{A}$ відбирається в домені NTT. Аналогічно, множення на відкритий ключ $\hat{t}$ повинні використовувати NTT, оскільки відкритий ключ передається в домені NTT. Як наслідок, реалізації також захочуть використовувати NTT для всіх інших множень, тому робимо ці виклики NTT і NTT^{-1} також явними в алгоритмах. Секретний ключ *sk* також зберігається в домені NTT.

Вартість цієї підвищеної простоти була б κ^2 додаткових операцій NTT як при генерації ключів, так і при інкапсуляції, що призведе до помітного уповільнення.

Для генерації публічної рівномірно випадкової матриці A було вирішено застосувати підхід NewHope «проти всіх повноважень». Це означає, що матриця не є системним параметром, а створюється заново як частина кожного відкритого ключа. Цей підхід має дві переваги: по-перше, це дозволяє уникнути дискусій про те, як саме був створений рівномірно випадковий системний параметр. По-друге, він захищає від сценарію атаки «все за ціною одного», коли зловмисник використовує серйозну кількість обчислень, щоб знайти коротку основу решітки, охоплену A, а потім використовувати цю коротку базу для атаки на всіх користувачів. Вартість такого рішення – це розширення матриці A з випадкового початкового рівня під час генерації та інкапсуляції ключа.

Біноміальний шум. Теоретичні розробки шифрування на основі LWE зазвичай розглядають LWE з гауссовим шумом, або округленим гауссовим, або дискретним гауссовим. У результаті багато ранніх реалізацій також відбирали шум із дискретного гаусового розподілу, який виявляється або досить неефективним або вразливим до атак на час. Ефективність найвідоміших атак на шифрування на основі LWE залежить не від точного розподілу шуму, а від стандартного відхилення (і, можливо, ентропії). Це мотивує використовувати розподіл шуму, з якого можно легко, ефективно та безпечно відбирати вибірки. Іншим прикладом є використання «навчання з округленням» (LWR), яке додає детермінований однорідний шум шляхом відкидання бітів, як у функції Kyber Compressq. У розробці Kyber вирішили використовувати центрований біноміальний шум і, таким чином, покладатися на LWE замість LWR як основну проблему.

Дозволяє помилки декапсуляції. Ще одне цікаве дизайнерське рішення полягає в тому, чи дозволити помилки декапсуляції (тобто помилки дешифрування в Kyber.CPAPKE) чи вибрати параметри, які мають не тільки незначну, але й нульову ймовірність збою. Переваги нульової ймовірності збою очевидні: CCA-перетворення та підтвердження безпеки стають легшими, і можно уникнути всього обговорення атак, які використовують помилки

декапсуляції. Недоліком розробки шифрування на основі LWE з нульовими збоями є те, що це означає або зниження безпеки від атак, спрямованих на основну проблему решітки (за рахунок значного зменшення шуму), або зниження продуктивності (за рахунок компенсації втрати безпеки за рахунок збільшення розміру решітки).). Рішення дозволити ймовірність збоїв у всіх наборах параметрів Kyber відображає інтуїцію:

- помилки декапсуляції є проблемою, якщо вони з'являються з невеликою ймовірністю; але
- атаки, які намагаються використати збої, які відбуваються з надзвичайно низькою ймовірністю, як у Kyber, є набагато меншою загрозою, ніж, наприклад, удосконалення гібридних атак схем націлювання з дуже низьким рівнем шуму.

Різні значення шуму η_1 і η_2. Зверніть увагу, що в Алгоритмах є додатковий неявний шум, створений за допомогою Compress_q. Це призводить до додавання певного (детермінованого) шуму до зашифрованого тексту, який можна інтерпретувати як збільшення шуму поліномів помилок e_1 та e_2. Якщо ймовірність помилки дешифрування досить низька, тоді має сенс також збільшити шум інших секретних термінів (тобто s; e; r), щоб бути на такому ж рівні, як e_1 плюс детермінований шум. Ця ідея використовується (виключно) для набору параметрів Kyber512. Покладатися на шум округлення від Compress_q, щоб додати помилку, схоже на припущення LWR. Але на відміну від схем (кільце/модуль)-LWR, де безпека повністю залежить від шуму, який детерміновано генерується округленням, наша залежність від детермінованого шуму набагато менша. По-перше, він додає лише 6 біт твердості Core-SVP, а по-друге, додаємо шум і округлення разом, що, ймовірно, має меншу алгебраїчну структуру, ніж просто округлення. Коротше кажучи, без припущення LWR, наш набір параметрів для Kyber512 має 112 біт твердості ядра SVP – точніше, відкриті ключі захищені 118 бітами, а шифротексти – 112; зі слабкою версією припущення LWR, він всюди має 118-бітну безпеку.

2.3.2 Алгоритм CRYSTALS-Dilithium

Схема цифрового підпису Dilithium, безпека якої заснована на жорсткості пошуку коротких векторів у ґратках. Ця схема була розроблена з урахуванням таких критеріїв:

Простий у безпечній реалізації. Найкомпактніші схеми сигнатур на основі решітки надзвичайно вимагають генерування секретної випадковості з дискретного гауссового розподілу. Генерування таких зразків у спосіб, захищений від атак з боку бічних каналів, дуже нетривіальний і може легко призвести до небезпечних реалізацій. Хоча можливо, що дуже ретельна реалізація може запобігти подібним атакам, нерозумно припускати, що універсально розгорнута схема, що містить багато тонкощів, завжди буде професійно реалізована. Тому дилітій використовує лише однорідну вибірку, як це було спочатку запропоновано. Крім того, всі інші операції (такі як поліноміальне множення та округлення) легко реалізуються за постійний час.

Dilithium прагне забезпечити довгострокову безпеку, було проаналізували застосовність решітчастих атак з дуже сприятливої для зловмисника точки зору. Зокрема, розглядатимемо квантові алгоритми, які потребують практично стільки ж місця, скільки часу. Такі алгоритми наразі нереальні, і, здається, існують серйозні перешкоди для усунення вимоги до місця, але допускається можливість покращення в майбутньому.

Мінімізуйте розмір відкритого ключа та підпису. Оскільки багато програм вимагають передачі як відкритого ключа, так і підпису (наприклад, ланцюжки сертифікатів), розроблена схема, щоб мінімізувати суму цих параметрів. Відповідно до обмеження, що уникаємо (дискретної) гаусової вибірки, Dilithium має найменшу комбінацію розмірів підпису та відкритого ключа серед будь-якої схеми на основі решітки з однаковими рівнями безпеки.

Dilithium є модульними і це надає йому легкість змінювати рівень безпеки. Дві операції, які становлять майже всю процедуру підписання та перевірки, — це розширення XOF (SHAKE-128 і SHAKE-256) і множення в кільці поліномів $Z_q[X]/(X^n + 1)$. Тому високоефективні реалізації нашого алгоритму потребують оптимізації цих операцій і забезпечення їх постійного виконання. Для всіх рівнів безпеки наша схема використовує те саме кільце з $q = 2^{23} - 2^{13} + 1$ і $n = 256$. Змінна безпека просто передбачає виконання більше/менш операцій над цим кільцем і виконання більшого/меншого розширення XOF. Іншими словами, як тільки отримано оптимізовану реалізацію для деякого рівня безпеки, отримати оптимізовану реалізацію для вищого/нижчого рівня майже тривіально.

Конструкція схеми заснована на підході «Фіат-Шамір з перервами». На рис. 1 зображено спрощену версію схему підпису для загального ознайомлення. Ця версія, по суті, є дещо зміненою схемою з [BG14]. Буде розглянено кожен з його компонентів, щоб дати уявлення про те, як працюють такі схеми.

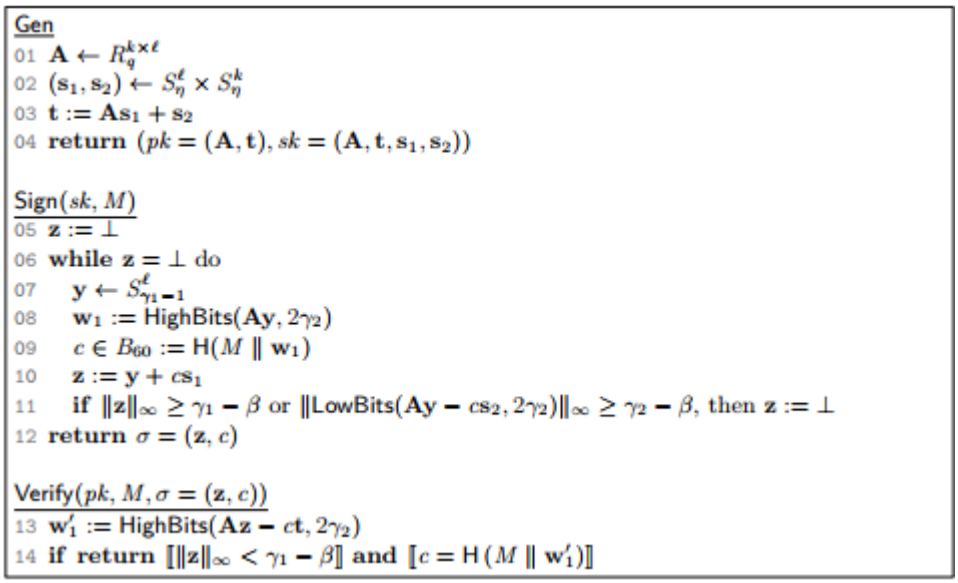

Рисунок 2.1 Шаблон схеми підпису Delithium.

Генерація ключів. Алгоритм генерації ключа генерує матрицю k ×` A, кожен з елементів якої є поліномом у кільці $R_q = Z_q[X]/(X^n + 1)$. $q = 2^{23} - 2^{13} + 1$

і $n = 256$. Після цього алгоритм відбирає випадкові вектори секретного ключа s_1 і s_2. Кожен коефіцієнт цих векторів є елементом R_q з малими коефіцієнтами - розміром не більше η. Нарешті, друга частина відкритого ключа обчислюється як $t = As_1 + s_2$. Вважається, що всі алгебраїчні операції в цій схемі виконуються над кільцем поліномів R_q.

Процедура підписання. Алгоритм підписання генерує вектор маскування поліномів y з коефіцієнтами менше γ_1. Параметр γ_1 встановлений стратегічно – він достатньо великий, щоб кінцевий підпис не розкривав секретний ключ (тобто алгоритм підписання має нульову інформацію), але досить малий, щоб підпис не було легко підробити. Потім підписувач обчислює Ay і встановлює w_1 як «старші» біти коефіцієнтів у цьому векторі. Зокрема, кожен коефіцієнт w у Ay можна записати канонічним способом як $w = w_1 \cdot 2\gamma_2 + w_0$, де $|w_0| \leq \gamma_2$; w_1 - це вектор, що містить усі w_1. Тоді виклик c створюється як хеш повідомлення та w_1. Вихід $\boldsymbol{c}$ є поліномом від R_q з рівно 60 ±1, а решта 0. Причина такого розподілу полягає в тому, що c має малу норму і походить з області розміром $> 2^{256}$. Потенційний сигнатур потім обчислюється як $z = y + cs_1$.

Якби z було виведено безпосередньо в цей момент, то схема підпису була б небезпечною через те, що секретний ключ був би витоком. Щоб уникнути залежності z від секретного ключа, використовуємо вибірку відхилення. Параметр β встановлюється як максимально можливий коефіцієнт csi. Оскільки c має 60 ± 1, а максимальний коефіцієнт в si дорівнює η, легко побачити, що $\beta \leq 60\eta$. Якщо будь-який коефіцієнт z більший за $\gamma_1 - \beta$, відхиляємо та перезапускаємо процедуру підписання. Крім того, якщо будь-який коефіцієнт молодших бітів $Az - ct$ більший за $\gamma_2 - \beta$. Перша перевірка необхідна для безпеки, а друга необхідна як для безпеки, так і для коректності. Цикл while у процедурі підписання повторюється до тих пір, поки не будуть задоволені дві попередні умови. Параметри встановлені таким чином, щоб

очікувана кількість повторів була не надто високою (у цих екземплярах це число становить від 4 до 7).

Перевірка. Верифікатор спочатку обчислює w_1` як старші біти Az−ct, а потім приймає, якщо всі коефіцієнти z менші за $\gamma_1 - \beta$ і якщо c є хешем повідомлення, а $w_1$`. Перевірка працює, зокрема, тому що HighBits(Az − ct, $2\gamma_2$) = HighBits(Ay, $2\gamma_2$). Перше, що слід помітити, це те, що Az − ct = Ay − cs_2.

$$\text{HighBits}(Az - ct, 2\gamma_2) = \text{HighBits}(Ay, 2\gamma_2). \quad (1)$$

Причина цього полягає в тому, що дійсний підпис матиме $||\text{LowBits}(Ay - cs_2, 2\gamma_2)||_\infty < \gamma_2 - \beta$. І оскільки знаємо, що коефіцієнти cs_2 менші за β, знаємо, що додавання cs_2 недостатньо для того, щоб викликати будь-які перенесення шляхом збільшення будь-якого коефіцієнта низького порядку, щоб мати величину щонайменше γ_2. Таким чином, рівняння (1) вірно, і підпис перевіряється правильно.

Основний шаблон на рис. 1, як і є, досить неефективний. Найбільш вражаюча (але легко виправна) неефективність полягає в тому, що відкритий ключ складається з матриці з k * *l* поліномів, які можуть мати досить велике представлення. Очевидним рішенням є створення A з деякого насіння ρ за допомогою SHAKE-128, і це стандартна техніка. Новизна Dilithium порівняно з попередніми схемами полягає в тому, що також зменшуємо розмір відкритого ключа в 2,5 раза за рахунок збільшення підпису приблизно на 150 байт. Для рекомендованого рівня безпеки схема має 2,7 КБ підписів і 1,5 КБ відкритих ключів.

Основне спостереження для отримання цього дуже сприятливого компромісу полягає в тому, що коли верифікатор обчислює w_1` в рядку 13, біти вищого порядку Az − ct не залежать надто від молодших бітів t, оскільки t множиться на a дуже маловагова поліном *c*. У схемі деякі молодші біти t не включені в відкритий ключ, і тому перевіряюча частина не завжди може правильно обчислити старші біти Az − ct. Щоб компенсувати це, підписувач

включає деякі «підказки» як частину підпису, які, по суті, є переносами, викликаними додаванням добутку *c* з відсутніми молодшими бітами t. За допомогою цієї підказки верифікатор може правильно обчислити w_1`.

Крім того, робимо нашу схему детермінованою, використовуючи стандартну техніку додавання початкового коду до секретного ключа та використання цього початкового елемента разом із повідомленням для отримання випадковості у у рядку 07. Недавній результат Kiltz et al. [KLS17] показав, що чим менше різних сигнатур бачить супротивник для одних і тих же повідомлень, тим сильніше скорочення моделі квантового випадкового оракула між схемою підпису та основними припущеннями про жорсткість. Хоча неясно, чи є покращена квантова атака для рандомізованих сигнатур, пропонуємо детерміновану версію як варіант за замовчуванням. Наша повна схема на рис. 2 також використовує основні оптимізації, такі як попереднє хешування повідомлення M, щоб не повторювати його при кожній спробі підписання.

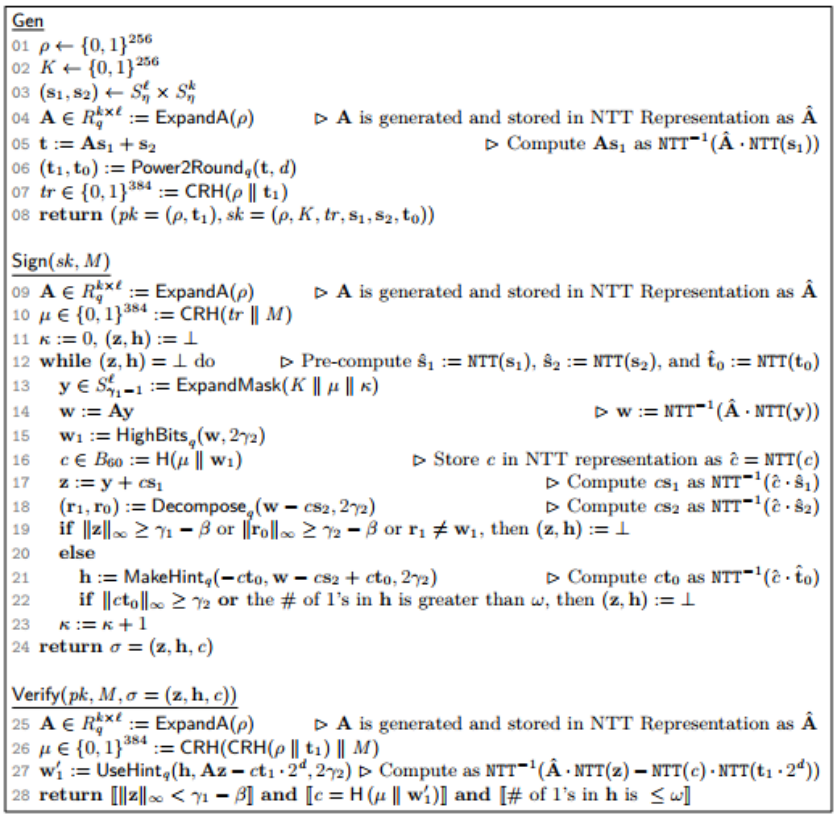

Рисунок 2.2 Схема підпису Dilithium.

Кільцеві операції. Позначимо R і R_q відповідно кільця $Z[X]/(X^n + 1)$ і $Z_q[X]/(X^n + 1)$, для q ціле число. У цьому документі значення n завжди буде 256, а q буде простим 8380417 = 223 − 213 + 1. Звичайні літери шрифту позначають елементи в R або R_q (що включає елементи в Z і Z_q) і напівжирні нижні літери представляють вектори-стовпці з коефіцієнтами в R або R_q. За замовчуванням усі вектори будуть векторами-стовпцями. Жирні великі літери є матрицями. Для вектора v позначимо через v^T його транспонування. Логічний оператор $[\![statement]\!]$ оцінює 1, якщо оператор істинний, і 0 в іншому випадку.

Модульні скорочення. Для парного (відповідно непарного) натурального числа α визначаємо $r' = r \bmod^{\pm} \alpha$ як єдиний елемент r' в діапазоні $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (відповідно $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) такий, що $r' \equiv r \bmod \alpha$. Іноді називатимемо це центрованим скороченням за модулем q.[2] Для будь-якого натурального числа α визначаємо $r' = r \bmod^{+} \alpha$ як єдиний елемент r' в діапазоні $0 \leq r' < \alpha$, такий що $r' \equiv r \bmod \alpha$. Коли точне представлення не важливе, пишемо $r \bmod \alpha$.

Розміри елементів. Для елемента $w \in Z_q$ пишемо $||w||_\infty$ як $|w \bmod^{\pm} q|$. Визначимо норми l_∞ і l_2 для $w = w_0 + w_1X + \ldots + w_{n-1}X^{n-1} \in R$:

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

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

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

будемо писати S_η, щоб позначити всі елементи $w \in R$ такі, що $||w||_\infty \leq \eta$.

Хешування. Схема використовує кілька різних алгоритмів, які хешують рядки в $\{0, 1\}^*$ на домени різних форм. Нижче буде наведено опис високого рівня цих функцій.

Нехай B_h позначає множину елементів R, які мають h коефіцієнтів, які дорівнюють -1 або 1, а решта дорівнюють 0. Маємо $|B_h| = 2^h * \binom{n}{h}$. Для нашої схеми підпису нам знадобиться криптографічна хеш-функція, яка хешує B_{60} (який має більше 2256 елементів). Алгоритм, який будемо використовувати для створення випадкового елемента в B_{60}, іноді називають версією «навиворіт» перетасування Фішера-Йейтса [Knu97], а його високорівневий опис наведено на рис. 2.3.

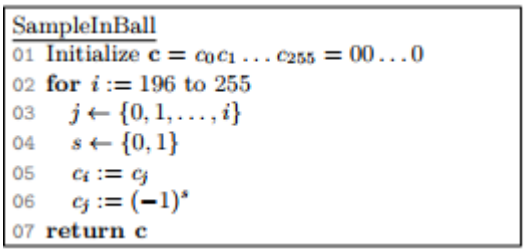
SampleInBall
01 Initialize $\mathbf{c} = c_0 c_1 \dots c_{255} = 00 \dots 0$
02 **for** $i := 196$ to 255
03 $j \leftarrow \{0, 1, \dots, i\}$
04 $s \leftarrow \{0, 1\}$
05 $c_i := c_j$
06 $c_j := (-1)^s$
07 **return c**

Рисунок 2.3 Створення випадкового масиву із 256 елементів та 60 ±1’ і 196 0’.

Розширення матриці A. Функція ExpandA відображає рівномірне насіння $\rho \in \{0, 1\}^{256}$ в матрицю $A \in R_q^{k*1}$ у представленні області NTT. Матриця A потрібна лише для множення. Отже, для швидших реалізацій функція розширення ExpandA не виводить $A \in R_q^{k*1} = (Z_q[X]/(X^{256} + 1))^{k\times l}$. Замість цього він виводить $\hat{A} \in Z_q^{256}$, що інтерпретується як представлення домену NTT для A. Оскільки вибірка A має бути рівномірною, а NTT є ізоморфізмом, ExpandA також потребує рівномірної вибірки в цьому поданні. Щоб бути сумісною з Dilithium, реалізація, чия NTT створює вектори, упорядковані інакше, ніж наш еталонний NTT, потребує вибірки коефіцієнтів у непослідовному порядку.

Вибірка векторів y. Функція ExpandMask, що використовується для детермінованого генерування випадковості схеми підпису, відображає K ||μ|| на $y \in S_{y_1-1}^l$.

Стійкий до зіткнень хеш. Функція CRH, яка використовується цій схемі підпису, є стійкою до зіткнень хеш-функцією, що відображається на $\{0, 1\}^{384}$.

Підпис. Алгоритми генерації ключів, підписання та перевірки для нашої схеми підпису представлені на рис. 4. Представляємо детерміновану версію схеми, в якій генерується випадковість, що використовується в процедурі підписання (за допомогою SHAKE-256), як детермінована функція повідомлення та невеликий секретний ключ. Оскільки нашу процедуру підпису, можливо, доведеться повторити кілька разів, поки не буде створено підпис, також додаємо лічильник, щоб зробити вихід SHAKE-256 різним при кожній спробі підпису одного й того самого повідомлення. Крім того, через те, що для підписання кожного повідомлення може знадобитися кілька ітерацій, обчислюємо початковий дайджест повідомлення, використовуючи стійку до зіткнень хеш-функцію, і використовуємо цей дайджест замість повідомлення протягом усієї процедури підписання.

Головне вдосконалення дизайну Dilithium порівняно зі схемою на рис. 1 полягає в тому, що розмір відкритого ключа зменшується приблизно в 2,5 раза за рахунок додаткових ста байтів у сигнатурі. Щоб досягти зменшення розміру, алгоритм генерації ключа виводить t_1 := Power2Roundq(t, d) як відкритий ключ замість t, як на рис. 1. Це означає, що замість [log q] біт на коефіцієнт, відкритий ключ вимагає [log q] − d бітів. У цьому прикладі $q \approx 2^{23}$ і $d = 14$, що означає, що замість 23 бітів у кожному коефіцієнті відкритого ключа є 9.

Основна проблема з відсутністю всього t у відкритому ключі полягає в тому, що алгоритм перевірки більше не може точно обчислити w_1' у рядку 13 на рис. 1. Для цього алгоритму перевірки знадобляться біти вищого порядку Az – ct, але він може обчислити лише $Az - ct_1 \cdot 2^d = Az - ct + ct_0$. Але оскільки добуток ct_0 складається лише з малих чисел, а дбаємо лише про біти вищого порядку, нам насправді потрібно знати лише перенесення, які викликає кожен коефіцієнт ct_0. Це носії, які підписувач надсилає як підказку перевіряючому.

Евристично, виходячи з нашого вибору параметрів, не повинно бути більше ω позицій, у яких викликається перенос. Таким чином, підписувач просто надсилає позиції, в яких відбуваються ці перенесення (це додаткові байти в сигнатурі), що дозволяє верифікатору обчислити старші біти $Az - ct$.

3 УМОВИ РЕАЛІЗАЦІЇ ТА ВИМОГИ ДО ЗАСТОСУВАННЯ АЛГОРИТМІВ ФОРМУВАННЯ ЦИФРОВОГО ПІДПИСУ

3.1 Порівняльний аналіз постквантових алгоритмів формування цифрового підпису.

NIST виділяє три основних критерії, за якими проводилася оцінка кандидатів: безпека, швидкодія та використання ресурсів пам'яті, характеристики алгоритмів та нюанси реалізацій. Безпека є найважливішою вимогою серед них. NIST планує використовувати новий постквантовий стандарт для виконання широкого ряду завдань. Оцінка кандидатів відбуватиметься саме за їхньою здатністю забезпечення безпеки у цих криптографічних завданнях. Схеми електронного підпису повинні бути стійкими до атак при шифртексті (adaptive chosen cipher text attack), атак на основі відомих повідомлень, до екзистенційної підробки та інших видів атак. [11]

NIST визначає 5 рівнів стійкості (категорій безпеки):

- 1 рівень – «ймовірно, безпечні алгоритми в найближчому майбутньому (якщо не виявиться, що КК працюватимуть швидше, ніж очікується)»;
- 2, 3 рівні – «ймовірно, безпечні алгоритми в найближчому майбутньому";
- 4, 5 рівні – «ймовірно, надмірні за стійкістю алгоритми»;

Також однією з вимог для кандидатів було надання інформації для узагальнення про відомі криптоаналітичні атаки на запропоновані схеми та оцінка складності цих атак.

Друга за значущістю вимога – вимога до ефективності виконання та використання ресурсів пам'яті. Сюди відносяться: - розміри ключів, розміри підпису; час, витрачений на генерацію ключів та підписи; час, потрібний на перевірку підпису; а також частка можливих помилок під час роботи алгоритмів. Вимоги за розміром пам'яті відносяться як до програмного, так і апаратного забезпечення.

Критерій «характеристики алгоритмів та нюанси реалізацій» полягають у будь-яких специфічних можливостях кожного алгоритму, наприклад, у хорошій здатності ефективно працювати на різних платформах, або у можливості паралелювання обчислень для досягнення більшої швидкодії. Розглянемо кандидатів другого туру серед схем електронного підпису.

Для схем підпису визначальним параметром є довжина підпису, а також довжина відкритого ключа підписувача. Розглянемо постквановані алгоритми цифрового підпису: NTRUEncrypt, CRYSTAL-Dilithium, CRYSTAL-Kyber (Таблиця 3.1). Параметри, за якими виконується порівняльний аналіз наступні: довжина закритого ключа, довжина відкритого ключа, довжина підпису.

Таблиця 3.1 Постквантові алгоритми цифрового піпдису.

Алгоритми	Конкретна реалізація	Довжина відкритого ключа	Довжина закритого ключа	Довжина підпису	Категорія безпеки
NTRUEncrypt	ntruhps2048509	935	699	699	1
	ntruhps2048677	1235	931	931	3
	ntruhps4096821	1592	1230	1230	5
CRYSTAL-Dilithium	Dilithium medium	2800	1184	2044	1

Продовження таблиці 3.1

CRYSTAL-Dilithium	Dilithium_recommended	3504	1472	2701	3
	Dilithium_very_high	3856	1760	3366	5
CRYSTAL-Kyber	Kyber-512	1632	800	768	1
	Kyber-768	2400	1184	1088	3
	Kyber-1024	3618	1568	1568	5

У таблиці 3.1 перелічено три різні набори параметрів, спрямовані на різні рівні безпеки. Зокрема, Kyber-512 націлений на безпеку, приблизно еквівалентну AES-128, Kyber-768 націлена на безпеку, приблизно еквівалентну AES-192, а Kyber-1024 націлена на безпеку, приблизно еквівалентну AES-256. [16]

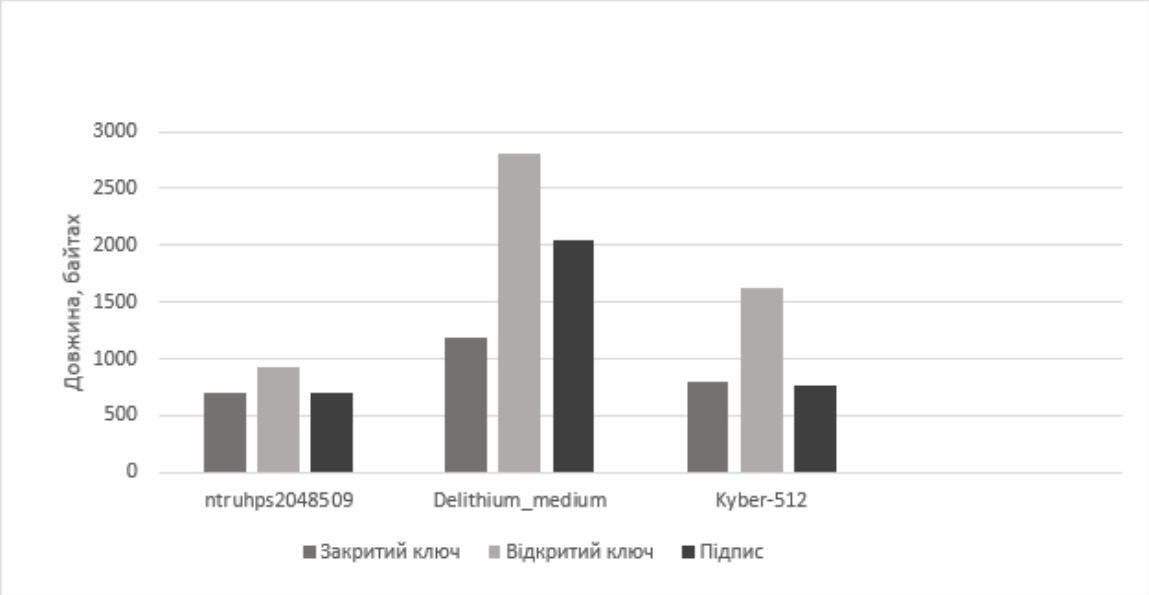

Рисунок 3.1 Довжини ключів та підписів при першому критерії безпеки.

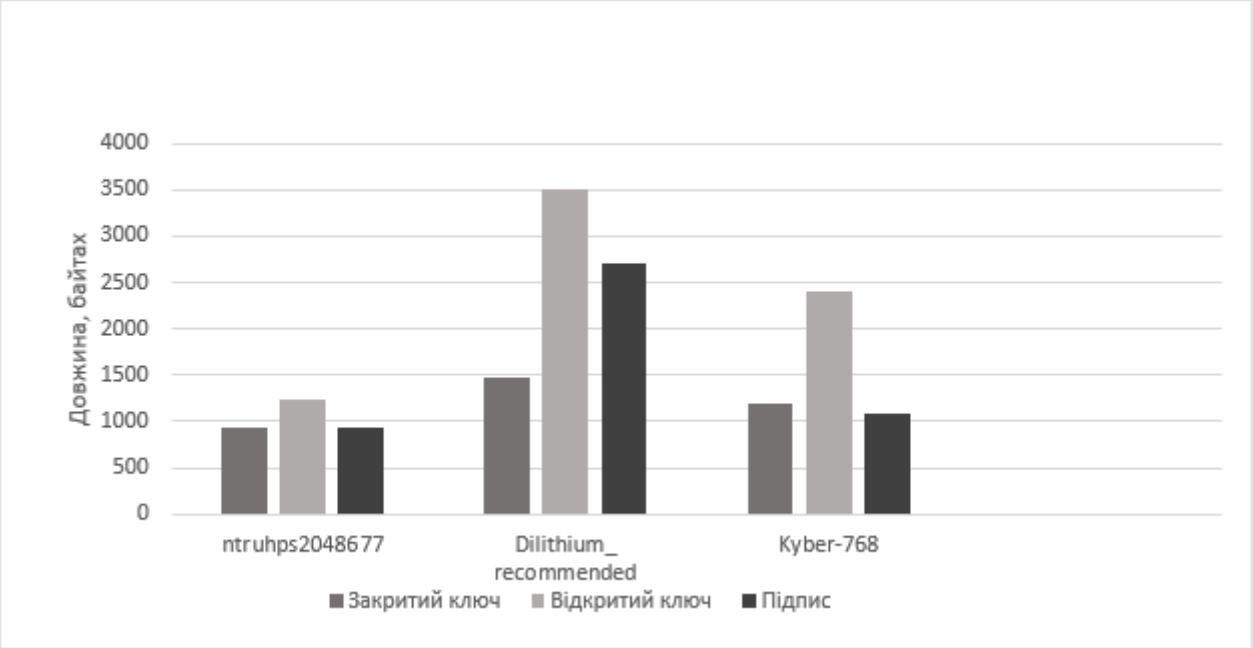

Рисунок 3.2 Довжини ключів та підписів при другому і третьому критерії безпеки.

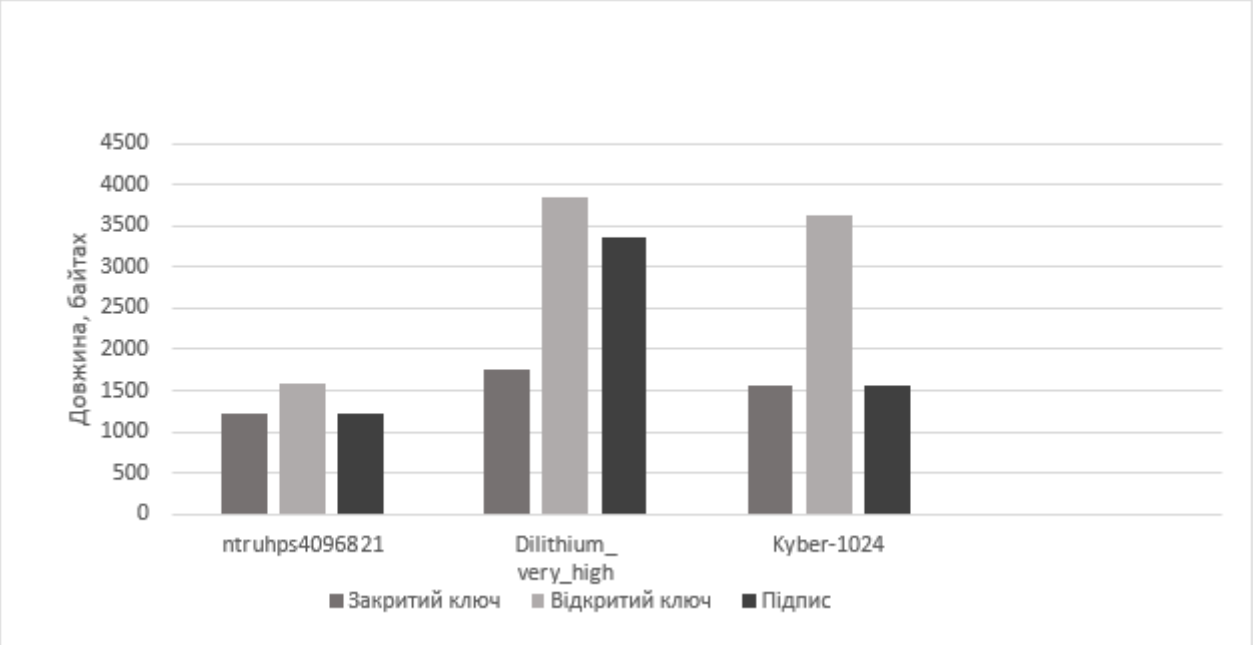

Рисунок 3.3 Довжини ключів та підписів при четвертому і п’ятому критерії безпеки.

Проведений порівняльний аналіз пост-квантових алгоритмів електронного підпису який виявив, що згідно рисунків 3.1, 3.2 і 3.3 найбільш

перспективний кандидат – алгоритм NTRUEncrypt, який за усіх критеріїв безпеки забезпечував найменшу довжину ключів та підпису.

3.2 Основні атаки на постквантові алгоритми цифрового підпису.

1) атака грубої сили, тобто повного перебору;

Брутфорсом називається метод злому облікових записів шляхом підбору паролів до них. Термін утворений від англомовного словосполучення "brute force", що означає в перекладі "груба сила". Суть підходу полягає у послідовному автоматизованому переборі всіх можливих комбінацій символів з метою рано чи пізно знайти правильну. [17]

З цієї точки зору пошук пароля можна розглядати як математичне завдання, вирішення якого знаходиться за досить великої кількості спроб. Програмне забезпечення для брутфорсу генерує варіанти паролів та перевіряє кожен із них. З погляду математичних методів, вирішити завдання в такий спосіб можна завжди, але тимчасові витрати на пошуки не завжди виправдовують мету, оскільки поле пошуку рішень величезне.

Брутфорс - один з найпопулярніших методів злому паролів до облікових записів онлайн-банків, платіжних систем та інших веб-сайтів. Втім, зі зростанням довжини пароля цей спосіб стає незручним, оскільки зростає час, що необхідно на перебір всіх можливих варіантів. Також за його допомогою можна перевіряти криптостійкість пароля. Брутфорс ще називають методом вичерпування, тому що правильна комбінація виявляється шляхом аналізу всіх можливих варіантів та відкидання кожного невідповідного поєднання.

Існує кілька видів атаки методом «грубої сили»:

- Персональний злам. У цьому випадку брутфорс спрямований на отримання доступу до особистих даних конкретного користувача:

облікових записів соціальних мереж, пошті, сайту. Під час спілкування через інтернет, у тому числі використовуючи шахрайські схеми, зловмисник намагається дізнатися про логін, персональні відомості та іншу інформацію, яка знадобиться для підбору пароля. Далі зломщик прописує в спеціальну програму адресу ресурсу, до якого потрібен доступ, логін облікового запису, підключає словник та підбирає пароль. Якщо пароль користувача заснований на особистій інформації і складається з малої кількості символів, спроба зловмисника може принести успіх навіть за короткий час.

- "Брут-чек". Цей вид брутфорсу означає полювання на паролі у великій кількості. Відповідно, мета - заволодіти даними не одного користувача, а безлічі різних облікових записів на декількох веб-ресурсах. До програми хакера підключається база логінів і паролів будь-яких поштових сервісів, а також проксі-лист, щоб замаскувати вузол, не давши веб-сервісам пошти виявити атаку. При реєстрації на сайті, у соціальній мережі або у грі користувач заповнює поле з адресою своєї пошти, на яку надходять дані для входу до відповідного облікового запису. В опціях брутфорсу прописується список назв сайтів або інших ключових слів, за якими програма шукатиме в поштових скриньках саме ці листи з логінами та паролями, вийматиме та копіюватиме інформацію в окремий файл. Так кіберзлочинець отримує сотні паролів і може використовувати їх у будь-яких цілях.
- Видалити злам операційної системи комп'ютерного пристрою. Брутфорс в комбінації з іншими зламують утилітами застосовується для отримання доступу до віддаленого ПК. Злом такого виду починається з пошуку мереж, які підходять для атаки. Адреси користувачів отримують спеціальними програмами або беруться з баз. Словники перебору та списки IP-адрес вводяться в налаштуваннях brute force. У разі успішного підбору пароля

зберігаються IP-адреса машини жертви та дані для входу, які далі використовуються зловмисником - наприклад, з метою повного керування ПК через утиліту Radmin або іншу подібну програму.

Цілі брутфорсу. Брутфорс дозволяє заволодіти доступом до облікових записів у соціальних мережах або онлайн-іграх, що може призвести до втрати конфіденційної інформації, цифрових валют, досягнень, потрапляння листування в чужі руки. З облікових записів може виконуватися розсилання спаму, здійснюватися вимагання та інші протиправні дії. Заволодівши великою кількістю облікових записів, хакер може їх обміняти або продати.

Отримання даних для входу до платіжних систем загрожує користувачам втратою грошових сум і навіть здобуттям боргів, оскільки зловмисник може вільно розпоряджатися фінансами, виконати переказ грошей, оформити кредит.

Підбір паролів до веб-сайтів методом brute force відкриває доступ до баз даних клієнтів, електронних адрес, використання майданчика з метою поширення шкідливих програм, розсилки спаму і т.п.

Отримавши точку входу у віддалену комп'ютерну систему за допомогою перебору паролів, зловмисник може виконувати різні злочинні дії від імені користувача, а також скористатися його особистими даними з метою шантажу, здирства, здійснити крадіжку секретної інформації та коштів.

Об'єктами впливу брутфорсу стають не тільки комп'ютери та облікові записи рядових користувачів інтернету, але й сайти, сервери, робочі станції комерційних та банківських структур, різних організацій.

Джерело загрози. Метод перебору паролів використовують кіберхулігани з метою зламати гру, пошту, обліковий запис у соцмережах. Зазвичай їхньою метою є заподіяння неприємностей іншим людям, перевірка своїх умінь, читання особистого листування.

Кіберзлочинці самі пишуть програми для злому або користуються результатами праці колег. Для перебору можуть використовуватися потужні комп'ютерні системи, у тому числі раніше зламані або орендовані. У руках зловмисників брутфорс є засобом отримання особистої вигоди з отримання доступу до облікових даних.

Також, як зазначалося, брутфорс може використовуватися з метою перевірки криптографічної стійкості паролів.

2) традиційна атака зустріч посередині;

У криптоаналізі методом зустрічі посередині або атакою "зустрічі посередині" (англ. meet-in-the-middle attack) називається клас атак на криптографічні алгоритми, що асимптотично зменшують час повного перебору за рахунок принципу "поділяй і владарюй", а також збільшення обсягу необхідної пам'яті. Вперше цей метод був запропонований Вітфілдом Діффі та Мартіном Хеллманом у 1977 році.

При спробі покращити безпеку блокового шифру приваблива ідея полягає в тому, щоб зашифрувати дані кілька разів за допомогою декількох ключів. Можна подумати, що це подвоює або навіть n-кортежів безпеку схеми множинного шифрування, залежно від того, скільки разів дані шифруються, тому що вичерпний пошук по всіх можливих комбінаціях ключів (простий перебір) вимагатиме 2 спроби. якщо дані зашифровані k-бітними ключами n разів.

MITM - це загальна атака, яка послаблює переваги безпеки використання кількох шифрів, зберігаючи проміжні значення із шифрів або дешифрування та використовуючи їх для скорочення часу, необхідного для перебору ключів дешифрування. Це робить атаку Meet-in-the-Middle (MITM) загальним просторово-часовим компромісом криптографічної атакою.

Атака MITM намагається знайти ключі, використовуючи як діапазон (зашифрований текст), так і домен (відкритий текст) композиції декількох

функцій (або блокових шифрів), так що пряме відображення через перші функції однакове як зворотне відображення (зворотне зображення) через останні функції, що буквально зустрічаються в середині складової функції. Наприклад, хоча Double DES шифрує дані двома різними 56-бітними ключами, Double DES можна зламати за допомогою 2 операцій шифрування та дешифрування.

Багатовимірний MITM (MD-MITM) використовує комбінацію декількох одночасних атак MITM, як описано вище, де зустріч відбувається в декількох позиціях складової функції. [18]

3) атака на основі алгоритму Arora-Ge;

Це атака через алгоритм Arora-Ge. Основна ідея полягає в тому, щоб розглянути зразок LWE (a, b := a^T s + e), де $e \in S \subseteq Z_q$, як поліноміальне рівняння

$$f_{a,b}(s) = \prod_{x \in R}(b - a^T s - x) \bmod q$$

де b, a відомі, а s розглядається як невідома змінна (позначається підкресленням). Очевидно, що якщо (a, b) є зразком LWE, то $f_{a,b}(s) = 0 \; mod \; q$, інакше це не так. Розв'язування системи поліноміальних рівнянь

$$\{f_{a_i,b_i}(s) = 0 \bmod q\}_{i=1}^{m}$$

ступеня |S| дасть нам секрет LWE.

Однак, розв'язування систем поліноміальних рівнянь (навіть рівнянь 2 ступеня) є NP-важким. Спостереження Арори та Ге полягають у тому, що якщо рівнянь достатньо багато, їх можна привести до лінійних, і що рішення отриманої лінійної системи дасть нам рішення поліноміальної системи w.h.p.

Ступінь поліномів дорівнює |S| (тобто область, в якій знаходяться члени помилки) і кількість одночленів, таким чином, дорівнює $\binom{n + |S|}{|S|}$. Зведення

до лінійних рівнянь - це основне перетворення, при якому кожен моном замінюється новою змінною. Крім того, якщо $m \gg \binom{n + |S|}{|S|}$, маємо більше рівнянь, ніж змінних. Почнемо з того, що будь-який розв'язок поліноміальної системи буде розв'язком лінеаризованої системи; отже, s є розв'язком. Коли m достатньо велике, також можемо показати, що s є єдиним рішенням. [19]

4) BKW атаки;

Алгоритм запропонований Blum, Kalai, Wasserman у 2003 році для вирішення задачі LPN (learning parity with noise) і може бути узагальнений на випадок задачі LWE. Алгоритм є модифікацією методу Гауса, використання якого у разі LWE неможливе через швидке накопичення помилки в процесі обчислень. BKW, як і метод Гауса, складається з трьох етапів: прямого ходу, знаходження значення останньої змінної та зворотного ходу. У першому етапі алгоритму BKW рядки підбираються те щоб кожен раз при відніманні відразу кілька значень зверталося в нуль. Це, така система рівнянь сильно перевизначена. Таким чином, за невелику кількість віднімань отримуємо блочну верхньо трикутну матрицю. Другий етап, «перевірка гіпотез», полягає у підборі останніх значень таких, щоб розподіл помилки був найближчим до очікуваного. Зворотний хід, як і алгоритмі Гауса, полягає у послідовної підстановці значень у рівняння. [20]

5) Primal attack (Search-LWE зводиться до BDD атаки);

Атака первинної решітки для LWE вирішує проблему декодування обмеженої відстані (BDD) [21] безпосередньо. Тобто, враховуючи вибірки LWE (A, b), він знаходить вектор $w = As$ такий, що $||b - w||$ надзвичайно малий. У літературі в основному розглядаються два підходи до вирішення BDD: перший безпосередньо вирішує BDD за допомогою найближчого алгоритму Бабая з подальшим зменшенням решітки [24], а другий перетворює екземпляр BDD в екземпляр (u)SVP і вирішує його за допомогою скорочення решітки. [3, 6]. Другий метод, який розглядається більш широко. Для цього методу дані

зразки LWE перетворюються в деяку решітку. Вкладення Каннана [25] розглядає форму ешелону стовпця $[I_n||A'^t]^t$ A $\in Z_q^{m\times n}$ (після відповідної перестановки рядків) і будує решітку Λ_{Kan}, породжену наступною матрицею

$$B_{Kan} = \begin{pmatrix} qI_{m-n} & A' \\ 0 & I_n \\ 0 & 0 \end{pmatrix} \begin{matrix} b \\ 1 \end{matrix} \in Z^{(m+1)*(m+1)},$$

яка має короткий вектор $(e, 1) \in Z^{m+1}$. Однак такий підхід не може бути корисним, коли секрет невеликий, і ця інформація може призвести до кращої атаки, дозволяючи зловмиснику використовувати її.

У зв'язку з цим, інше вбудовування решітки пропонує [9]:

$$A_{BG} = \{x \in Z^m * (vZ)^n * Z\} : \left(I_m || \frac{1}{v} || A - b\right) * x = \text{mod q}\}$$

Ця решітка містить незвичайний короткий вектор (*e*, *vs*, 1). Таким чином, можемо знайти секретний вектор s разом із вектором помилки *e*, розв'язавши SVP на решітці, створеній базою

$$B_{BG,v} = \begin{pmatrix} qI_m & A & -b \\ 0 & vI_n & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Коефіцієнт масштабування *v* визначається так, щоб короткий вектор (*e*, *vs*, 1) був збалансованим або явно

$$\|e\| \approx \|vs\|$$

При виборі такого *v* вектор (*e*, *vs*, 1) вважається таким, що має вигляд (e', 1), де e' відбирається з гауссового розподілу, що має таке ж стандартне відхилення як і *e*. [21]

6) Dual attack (Decision-LWE зводиться до SIS);

Подвійна атака. Подвійна атака полягає у знаходженні короткого вектору в подвійній решітці $w \in \Lambda' = \{(x, y) \in Z^m * Z^{kn} : A^t x\} = y \bmod q$. Припустимо, що знайшли вектор (x, y) довжини *l* і обчислюємо $z = v^t * b = v^t As$

+ $v^te = w^ts + v^te \bmod q$, який розподіляється як гауссове значення стандартного відхилення l_ς, якщо (A, b) є справді вибірка LWE (інакше це однорідний mod q). Ці два розподіли мають максимальну відстань варіації, обмежену $\epsilon = 4\exp(-2\pi^2\tau^2)$, де $\tau = l_s/q$, тобто за такої довжини вектору один має перевагу перед рішенням-LWE. [23]

Довжина ‘ вектору, заданого алгоритмом BKZ, визначається як ‘ = kb0k. Знаючи, що Λ0 має розмірність d = m + kn і об’єм qkn, отримуємо ‘ = δd−1qkn=d. Тому для отримання результата потрібно запустити BKZ з розмірністю блоку b, де

$$-2\pi^2\tau^2 \geq \ln(\epsilon/4).$$

Зазначимо, що невеликі переваги не мають значення, оскільки узгоджений ключ хешується: зловмиснику потрібна перевага щонайменше 1/2, щоб значно зменшити простір пошуку узгодженого ключа. Тому він повинен збільшити свою ймовірність успіху, побудувавши приблизно $1/\epsilon^2$. Оскільки алгоритми просіювання забезпечують вектори $2^{0.2075b}$, атака повинна повторюватися принаймні R разів, де

$$R = \max 1{,}1/(2^{0.2075b}\epsilon^2)$$

Це робить консервативне припущення, що всі вектори, надані алгоритмом, такі ж короткі, як і найкоротший.

Атаки проти симетричних примітивів. Усі симетричні будівельні блоки Kyber створені за допомогою функцій, похідних від Keccak. У детермінованому розкладанні A з ρ нам, по суті, потрібен алгоритм SHAKE-128 для отримання результату, який «виглядає рівномірно випадковим» і не створює жодних бекдорів у основній задачі решітки. У генерації шуму вимагаємо, щоб конкатенація секретного та загальнодоступних вхідних даних і передача цієї конкатенації на SHAKE-256 як вхід приводила до безпечної псевдовипадкової функції. Порушення будь-якої з цих властивостей SHAKE

було б великим проривом у криптоаналізі SHAKE, який вимагав би заміни SHAKE всередині Kyber.

Модель доказів безпеки SHAKE-128, SHA3-256 і SHA3-512 як випадкові оракули, тобто вони підпадають під дію стандартних обмежень доказів у моделі (квантового) випадкового оракула. Перетворення цих обмежень на атаку, що використовує екземпляр XOF, H або G за допомогою SHAKE і SHA3, знову стане великим проривом у розумінні доказів Keccak або випадкових оракулів загалом.

7) Атаки проти основної проблеми MLWE.

MLWE як LWE. Найвідоміші атаки на основну проблему MLWE в Kyber не використовують структуру в решітці. Тому при аналізі стійкості розв'язання проблеми MLWE як проблеми LWE. Обговоримо поточний стан техніки алгебраїчних атак, тобто атак, які використовують структуру модульних решіток (або ідеальних решіток) в кінці цього підрозділу. [23]

Атаки проти LWE. Існує багато алгоритмів для вирішення LWE, але багато з них не мають відношення до набору параметрів визначених в даній роботі. Зокрема, тому що є лише

$$m = (k + 1)n$$

Зразки LWE, доступні зловмиснику, ми можемо виключити тип атак BKW та атаки лінеаризації. Це, по суті, залишає перед нами дві атаки BKZ, які зазвичай називають первинними та подвійними атаками.

Алгоритм BKZ продовжує зменшувати решітковий базис за допомогою оракула SVP у меншому вимірі b. Відомо, що кількість викликів до цього оракула залишається поліноміальною, але конкретна оцінка кількості викликів є досить і сьогодні ще до кінця не вирішеною, і це підпорядковується новим евристичним ідеям.

Усе починається з аналізу, який ігнорує цей поліноміальний фактор, тобто враховує лише вартість одного виклику оракула SVP у вимірі b. Наразі також використовується дуже проста оцінка вартості твердості SVP. Ця методологія стійкості core-SVP була введена, як простий спосіб оцінки безпеки. У світлі криптоаналітичного прогресу, який відбувся під час 1 і 2 раундів оцінки NIST, метод залишається інформативним, але занадто грубим, щоб отримати точні оцінки безпеки, особливо для захисту від класичних зловмисників.

8) Критпоаналітичні атаки

Існує чотири основні типи криптоаналітичних атак.

1. Криптоаналітична атака за наявності лише відомого шифртексту.

Криптоаналітик має тільки шифртексти C_1, C_2, …, C_i кількох повідомлень, причому всі вони зашифровані з використанням одного і того ж алгоритму шифрування E^k. Робота криптоаналітика у тому, щоб розкрити вихідні тексти M_1, M_2, …, M_i . Чим більше повідомлень тим більше шансів обчислити ключ D^K, використаний для шифрування цих повідомлень. Цей варіант відповідає моделі зовнішнього порушника, який має фізичний доступом до лінії зв'язку, але має доступ до апаратури шифрування та дешифрування.

2. Криптоаналітична атака за наявності відомого відкритого тексту.

Криптоаналітик має доступ не тільки до шифртекстів C_1, C_2, …, C_i та кількох повідомлень, але також до відкритих текстів M_1, M_2, …, M_i цих повідомлень. Його робота полягає в знаходженні ключа D_K, який використовується при шифруванні цих повідомлень, або алгоритму розшифрування будь-яких нових повідомлень, зашифрованих тим самим ключем. причому всі вони зашифровані з використанням того самого алгоритму шифрування E^k. Можливість проведення такої атаки складається при шифруванні стандартних документів, що готуються за стандартними

формами, коли певні блоки даних повторюються та відомі. Він також застосовується при використанні режиму глобального шифрування, коли вся інформація на вбудованому магнітному носії записується у вигляді шифртексту, включаючи головний кореневий запис, завантажувальний сектор, системні програми та ін. інформації та отримати великий обсяг відомого вихідного тексту для виконання криптоаналізу.

3. Криптоаналітична атака за можливості вибору відкритого тексту.

Крипто-аналітик не тільки має доступ до шифртекстів C_1, C_2, …, C_i та пов'язаних з ними відкритих текстів M_1, $M2$, …, M_i цих повідомлень, але й може за бажанням вибирати відкриті тексти, які потім отримує у зашифрованому вигляді. Такий криптоаналіз виходить потужнішим порівняно з криптоаналізом з відомим відкритим текстом, тому що криптоаналітик може вибрати для шифрування такі блоки відкритого тексту, які дадуть більше інформації про ключ. Робота криптоаналітика полягає в пошуку ключа K, використаного для шифрування повідомлень, або алгоритму рашифрування Dk нових повідомлень, зашифрованих тим самим ключем. Цей варіант атаки відповідає моделі внутрішнього порушника.

4. Криптоаналітична атака з адаптивним вибором відкритого тексту.

Це особливий варіант атаки з вибором відкритого тексту. Криптоаналітик може не тільки вибирати відкритий текст, який потім шифрується, але змінювати свій вибір залежно від результатів попереднього шифрування. При криптоаналіз з простим вибором відкритого тексту криптоаналітик зазвичай може вибирати кілька великих блоків відкритого тексту для їх шифрування; при криптоаналіз з адаптивним вибором відкритого тексту він має можливість вибрати спочатку дрібніший пробний блок відкритого тексту, потім вибрати наступний блок залежно від результатів першого вибору, і т.д. Ця атака надає криптоаналітики ще більше можливостей, ніж попередні типи атак. [22]

3.3 Умови реалізації постквантових алгоритмів цифрового підпису

Грунтуючись на перераховані вище атак можна сказати, що при реалізації постквантових алгоритмів цифрового підпису повинні виконуватись такі умови:

1) надійність математичної бази, що застосовується для цифрових підписів при криптоперетвореннях;
2) практична захищеність криптографічних перетворень типу цифрових підписів від відомих атак;
3) реальна захищеність цифрових підписів від усіх відомих і потенційно можливих критпоаналітичних атак;
4) стійкість цифрових підписів в умовах появи квантового комп'ютера.

В кваліфікаційної роботі на основі порівняльного аналізу алгоритмів було з’ясовано, що надійність математичної бази притаманна варіаціям алгоритмів CRYSTAL та NTRU. Щодо практичної захищеності до атак, наведених в даній роботі, обидва алгоритми з запропонованими параметрами забезпечують 1-3 рівні безпеки, які визначені NIST, а алгоритм NTRU має набір параметрів для 5 рівня безпеки. Стійкість вказаних алгоритмів базується на задачі пошуку короткого вектору решітки, яка відноситься до NP задачі. Щодо реальної захищеності алгоритмів до потенційно існуючих атак, питання вимагає проведення додаткового дослідження відносно часових атак та атак витоку по побічним каналам. Однак, згідно проведеного аналізу щодо існуючих атак алгоритм NTRU має найкращі показники та умови реалізації для застосування в різних додатках та може замінити поширену криптосистему RSA.

ВИСНОВОК

Пройшла ера RSA та еліптичних кривих, спричинена появою великомасштабних квантових комп'ютерів, які пропонують великі перспективи науці та суспільству, але несуть із собою значну загрозу нашій глобальній інформаційній інфраструктурі.

Популярні криптографічні схеми такі як RSA та криптографія з еліптичною кривою, легко піддаватимуться злому квантовим комп'ютером. Це швидко прискорить старіння наших систем безпеки, які зараз розгорнуті, і серйозно впливатиме на будь-яку галузь, де потрібно зберігати інформацію в безпеці.

Тепер перед суспільством стоїть питання стандартизації і для цього NIST оголосив конкурс на найкращий постквановий алгоритм, для того, щоб вибрати та стандартизувати найкращі з них у своєму роді.

У кваліфікаційній роботі було розглянуто пост-квантові схеми цифрового підпису, що пройшли у другий тур конкурсу NIST на створення нових пост-квантових стандартів.

Проведено порівняльний аналіз за такими параметрами: довжина закритого ключа, довжина відкритого ключа, довжина підпису.

Були обрані такі кандидати: NTRUEncrypt, CRYSTAL-Delithium, CRYSTAL-Kyber. Відповідно до аналізу найкращим кандидатом на вивчення надалі є – NTRUEncrypt (в усіх трьох конкретних критеріях) має найменшу довжину з усіх представлених вище алгоритмів при порівнянному часі роботи та рівні безпеки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Lyubashevsly V (2009) Fiat-Shamir with aborts: Applications to Lattice and Factoring-Based Signatures. International Conference on the Theory and Application of Cryptology and Information Security – ASIACRYPT (Springer), P. 598-616.
2. Перспективні методи та засоби криптографічних перетворень [Електронний ресурс] – Режим доступу до ресурсу: https://openarchive.nure.ua/bitstream/document/15487/1/202_PI_5-28.PDF
3. Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology [Електронний ресурс] – Режим доступу до ресурсу: https://www.researchgate.net/publication/2893927_Analysis_and_Improvements_of_NTRU_Encryption_Paddings
4. Falcon – A Post-Quantum Signature Scheme [Електронний ресурс] – Режим доступу до ресурсу: https://pqshield.com/falcon-a-post-quantum-signature-scheme/
5. On the security of UOV [Електронний ресурс] – Режим доступу до ресурсу: https://eprint.iacr.org/2009/483.pdf
6. Аналіз основних існуючих пост-квантових підходів і схем електронного підписку [Електронний ресурс] – Режим доступу до ресурсу: https://cyberrus.com/wp-content/uploads/2019/07/58-68-230-19_8.-Komarova.pdf
7. Martin R Albrecht, Benjamin R Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the fLWE, NTRUg schemes! In International Conference on Security and Cryptography for Networks, pages 351–367. Springer, 2018.
8. Eurocrypt 2010 [Електронний ресурс] – Режим доступу до ресурсу: https://www.iacr.org/conferences/eurocrypt2010/

9. National Institute of Standards and Technology (2016) Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. [Електронний ресурс] – Режим доступу до ресурсу: https://csrc.nist.gov/CSRC/media/Projects/PostQuantumCryptography/documents/call-for-proposals-final-dec-2016.pdf
10. Guo Q, Johansson T, Nilsson A (2020) A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM. Cryptology ePrint Archive preprint. [Електронний ресурс] – Режим доступу до ресурсу: https://eprint.iacr.org/2020/743
11. Основні типи криптоаналітичних атак [Електронний ресурс] – Режим доступу до ресурсу: http://csaa.ru/osnovnye-tipy-kriptoanaliticheskih-atak/
12. Scott Fluhrer. Cryptanalysis of ring-LWE based key exchange with key share reuse. IACR Cryptology ePrint Archive report 2016/085, 2016. [Електронний ресурс] – Режим доступу до ресурсу: https://eprint.iacr.org/2016/085.
13. CRYSTALS-Kyber Algorithm Specifications And Supporting Documentation (version 3.01) [Електронний ресурс] – Режим доступу до ресурсу: https://pq-crystals.org/kyber/data/kyber-specification-round3-20210131.pdf
14. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange–a new hope. In Proceedings of the 25th USENIX Security Symposium, pages 327–343. USENIX Association, 2016. [Електронний ресурс] – Режим доступу до ресурсу: http://cryptojedi.org/papers/#newhope.
15. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, Advances in Cryptology – EUROCRYPT 2012, volume 7237 of LNCS, pages 719–737. Springer, 2012. [Електронний ресурс] – Режим доступу до ресурсу: http://www.iacr.org/archive/eurocrypt2012/72370713/72370713.pdf
16. Cryptographic Suite for Algebraic Lattices [Електронний ресурс] – Режим доступу до ресурсу: https://pq-crystals.org/kyber/index.shtml

17. Brute force [Електронний ресурс] – Режим доступу до ресурсу: https://www.anti-malware.ru/threats/brute-force
18. Метод зустрічі посередині [Електронний ресурс] – Режим доступу до ресурсу: https://studme.org/190626/informatika/metod_vstrechi_poseredine
19. The Learning with Errors Problem: Algorithms [Електронний ресурс] – Режим доступу до ресурсу: https://people.csail.mit.edu/vinodv/CS294/lecture2.pdf
20. International Journal of Open Information Technologies ISSN: 2307-8162 vol. 9, no.3, 2021 [Електронний ресурс] – Режим доступу до ресурсу: http://injoit.org/index.php/j1/article/viewFile/1082/1043
21. Revisiting the Hybrid attack on sparse and ternary secret LWE [Електронний ресурс] – Режим доступу до ресурсу: https://eprint.iacr.org/2019/1019.pdf
22. Основні типи криптоаналітичних атак [Електронний ресурс] – Режим доступу до ресурсу: http://csaa.ru/osnovnye-tipy-kriptoanaliticheskih-atak/
23. Hybrid Dual Attack on LWE with Arbitrary Secrets [Електронний ресурс] – Режим доступу до ресурсу: https://eprint.iacr.org/2021/152.pdf
24. Lindner, R., Peikert, C.: Better key sizes (and attacks) for lwe-based encryption. In: Proc. of CT-RSA' 11. vol. 65–58, pp. 319–339. Springer (2011)
25. Kannan, R.: Minkowski's convex body theorem and integer programming. Mathematics of operations research 12(3), 415–440 (1987)