УДК 004.056.5 **DOI:10.30837/rt.2023.1.212.06**

С.О. КАНДІЙ

АНАЛІЗ БЕЗПЕКИ ПЕРСПЕКТИВНИХ МЕХАНІЗМІВ ІНКАПСУЛЯЦІЇ КЛЮЧІВ У МОДЕЛІ CORE-SVP

Вступ

Криптографія на решітках є сучасним напрямком досліджень у постквантовій криптографії. Серед фіналістів третього етапу конкурсу NIST PQC [2] більшість перетворень грунтується на структурованих решітках. Зокрема, механізм інкапсуляції ключів CRYSTALS-Kyber [3], який був рекомендований до стандартизації, також грунтується на решітках (проблема Module-LWE [4]). У той же час, в Україні є діючий стандарт ДСТУ 8961:2019 [1], який грунтується на добре відомій проблемі NTRU [5], найкращі методи криптоаналізу якої використовують редукцію решіток.

Оцінка складності редукції решіток для криптографічних схем є давньою проблемою. Асимптотичні оцінки сильно відрізняються від експериментальних значень, тому для вирішення практичних задач був розроблений ряд евристичних методів. По-перше, стандартним засобом при роботі з редукцією решіток є модель core-SVP [6], згідно з якою вартість атаки є вартістю виклику SVP оракула у алгоритмі BKZ [7]. По-друге, робляться евристичні припущення щодо того, як буде змінюватися форма базису протягом процесу редукції. По-третє, робляться еврістичні припущення щодо умови успіху атаки.

За останні роки зроблений значний прогрес у криптографії на решітках. Метою цієї роботи є оцінка безпеки механізмів інкапсуляції ключів CRYSTALS-Kyber та ДСТУ 8961:2019 від прямих атак у моделі coreSVP з врахуванням останніх досягнень в редукції решіток. Для аналізу було обрано дві популярні евристики – GSA [6] та симулятор Чена – Нгуєна [8].

1. Відомості з теорії решіток

Введемо необхідні позначення з теорії решіток [7]. Решітка L з базисом B є множиною цілочисельних комбінацій лінійно незалежних векторів $b_1,...,b_n$: $L(B=b_1,...,b_n)=\left\{\sum_{i=1}^{n}x_ib_i, x_i\in Z\right\}$. Довжиною вектора v є стандартна евклідова норма $\|v\|=\sqrt{v\bullet v}$, де операція $\bullet$ є скалярним добутком та для двох векторів $v=(v_1,...,v_n)$ і $w=(w_1,...,w_n)$ визначена як $v\bullet w=\sum_{i=1}^{n}v_iw_i$. Для чисел i та j з $i<j$ позначення $[i:j]$ означає множину $\{i,i+1,...,j\}$. Для $[1:j]$ використовується позначення $[j]$.

Гамма функція $\Gamma(s)$ визначена для $s>0$ як $\Gamma(s)=\int_0^\infty t^{s-1}e^{-t}dt$. $Ball_n(R)$ означає n-вимірний шар радіуса R та його об'єм $V_n(R)=R^n\pi^{n/2}/\Gamma(n/2+1)$. Для заданного базису $B=(b_1,...,b_n)$ ортогоналізований за Граммом – Шмідтом базис є $B^*=(b_1^*,...,b_n^*)$, де $b_i^*=b_i-\sum_{j=1}^{i-1}\mu_{ij}b_j^*$ для $1\le j<i\le n$, де $\mu_{ij}=(b_i\bullet b_j^*)/\left\|b_j^*\right\|^2$ – коефіцієнти Грамма – Шмідта (ГШ-коефіцієнти), $\left\|b_j^*\right\|$ – довжини векторів Грамма – Шмідта (ГШ-довжини). Детермінант решітки визначений як $\det(L)=\prod_{i=1}^{n}\left\|b_i^*\right\|$ і дорівнює об'єму фундаментального паралелепіпеда $vol(L)$. Ортогональна проекція є відображення $\pi_i: R^n\mapsto span(b_i,...,b_{i-1})^\perp$ для $i\in\{1,...n\}$.

Локальний блок $L_{[i:j]}$ – (проективна) решітка, яка задається наступним чином:

$$L_{[i:j]} = B_i = L(\pi_i(b_i), \pi_i(b+1), ..., \pi_i(b_j)) \quad (1)$$

для $j \in \{i, i+1, ..., n\}$.

У кожній решітці L існує найменший ненульовий вектор. $\lambda_1(L)$ – норма найменшого вектора. Проблема пошуку найменшого вектора (SVP) полягає у пошуку векотора довжини $\lambda_1(L)$. Проблема апроксимації найменшого вектора γ-SVP полягає у пошуку вектора, що має норму, шо менша за $\gamma(n)\lambda_1(L)$, де n - розмірність решітки.

При аналізі часто використовується евристика Гаусса. Якщо задана n-вимірна решітка L та деяка область S, то кількість точок у $S \bigcap L$ приблизно $vol(S)/vol(L)$. Якщо взяти у якості S шар з радіусом R, то число точок можливо апроксимувати як $V_n(R)/vol(L)$, звідки можливо апроксимувати λ_1 як

$$\lambda_1(L) \approx \det(L)^{(1/n)} / V_n(1)^{(1/n)} = \frac{(\Gamma(n/2+1)\det(L))^{1/n}}{\sqrt{\pi}}. \quad (2)$$

Ця оцінка зветься евристикою Гаусса і позначається як $GH(L)$.

1.1. Складні проблеми в теорії решіток

Безпека криптографії на решітках переважно грунтується на проблемах NTRU та LWE (і її різновидах).

Проблема LWE. Нехай $n, q > 0$ – цілі числа, χ – деякий розподіл ймовірностей над множиною цілих чисел Z та s – секретний вектор з рівномірного розподілу над Z_q^n. $L_{s,\chi}$ є розподілом ймовірностей над $Z_q^n \times Z_q$, який отримується наступним чином. Обирається вектор $a \in Z_q^n$ з рівномірного розподілу, значення помилки $e \in Z_q$ з розподілу χ та повертається пара $(a, c) = (a, a \bullet s + e) \in Z_q^n \times Z_q$. Проблема LWE (обчислювальна версія) полягає у тому, щоб для поліноміальної кількості пар (a, c) з розподілу $L_{s,\chi}$ знайти вектор s.

Проблема NTRU. Нехай $n, q > 0$ – цілі числа і задано кільце R_q (на практиці – поле, проте у загальному випадку NTRU визначається для довільних кілець) поліномів ступеня n над кільцем лишків за модулем q. Нехай $f, g \in R_q$ – поліноми з деякого розподілу χ і $h = g/f$. Проблема NTRU (обчислювальна версія) полягає у пошуку поліномів f, g для заданного полінома h.

Проблема LWE має багато узагальнень. Для подальшого аналізу варто згадати про проблеми Ring-LWE [9] та Module-LWE [4]. Загальне визначення цих проблем потребує введення ряду понять з алгебраїчної теорії чисел, тому для простоти викладення наведемо визначення для випадку поля $R_q = Z_q[X]/(X^n+1)$, яке використовується у переважній більшості криптографічних схем на решітках.

Проблема Ring-LWE. Нехай $n, q > 0$ – цілі числа і задано поле $R_q = Z_q[X]/(X^n+1)$. χ – деякий розподіл ймовірностей над R_q та s – секретний поліном з рівномірного розподілу над R_q. $L_{s,\chi}$ є розподілом ймовірностей над $R_q \times R_q$, який отримується наступним чином. Обирається поліном $a \in R_q$ з рівномірного розподілу, поліном помилки $e \in R_q$ з розподілу χ

та повертається пара $(a,c)=(a,a\bullet s+e)\in R_q\times R_q$. Проблема Ring-LWE (обчислювальна версія) полягає у тому, щоб для поліноміальної кількості пар (a,c) з розподілу $L_{s,\chi}$ знайти поліном s.

Проблема Module-LWE визначена аналогічним чином, тільки a,s є векторами поліномів: $a,s\in(R_q)^d$, де $d>0$ деяке ціле число.

1.2. Зауваження щодо Module-LWE

Найкращими відомими методами вирішення проблем Module-LWE та LWE є атаки на основі редукції решіток. Проте, решітки для Module-LWE відрізняються від звичайних LWE решіток додатковою структурованістю, вони складаються з блоків фіксованого розміру. Множина всіх можливих блоків має бієктивне відображення на множину поліномів кільця $R_q=Z_q[X]/\phi(X)$ для деякого незвідного полінома $\phi(X)$. Для довільного кільця $Z_q[X]/\phi(X)$ відображення $rot(f)$ задається як

$$f\in R_q\mapsto\begin{pmatrix} f \bmod \phi(x)\\ f\cdot x \bmod \phi(x)\\ \vdots\\ f\cdot x^{n-1} \bmod \phi(x)\end{pmatrix}\in Z_q^{n\times n}. \tag{3}$$

Відповідно, кожен ідеал $\langle f_1,\ldots,f_{m-1}\rangle\in R_q$ може бути розглянутий як решітка з базисом:

$$\begin{pmatrix} rot(f_1)\\ \vdots\\ rot(f_m)\end{pmatrix}. \tag{4}$$

Решітки такого типу мають назву ідеальних решіток (ideal lattice) і проблема пошуку найменшого вектора (SVP) на них має назву Ideal-SVP.

Модуль M розмірності k над полем R_q задається як множина векторів вигляду $(g_1,\ldots,g_k)$, де кожен g_i належить до певного ідеалу $I_i\in R_q$. Аналогічно до ідеалів кожен модуль може бути розглянутий як решітка з базисом:

$$\begin{pmatrix} rot(f_{11}) & rot(f_{12}) & \cdots & rot(f_{1k})\\ rot(f_{21}) & rot(f_{22}) & \cdots & rot(f_{2k})\\ \vdots & \vdots & \ddots & \vdots\\ rot(f_{m1}) & rot(f_{m2}) & \cdots & rot(f_{mk})\end{pmatrix}. \tag{5}$$

Такі решітки є модульними решітками і проблема пошуку найменшого вектора на них має назву Module-SVP. Для вирішення проблеми Module-LWE зазвичай роблять зведення до проблеми Module-SVP. Будується модульна решітка, достатньо малий вектор якої містить інформацію про вирішення системи рівнянь Module-LWE.

З визначення модульних решіток видно, що вони мають доволі складну алгебраїчну структуру, що задає структурованість базису решітки. Розуміння впливу цієї структурованості на складність пошуку малих векторів є ключовим питанням для оцінки складності криптоаналізу.

Втім, методів використання структурованості модульних решіток існує небагато. Більшість робіт присвячені ідеальним решіткам і не можуть бути узагальнені на випадок модульних решіток.

Для проблеми Ideal-SVP відомі деякі результати [10-12] для квантових комп'ютерів. Зокрема, якщо ідеал є головним (має лише один твірний елемент), то алгоритм [12] дозволяє вирішити задачу Ideal-SVP з фактором апроксимації $\exp(O(\sqrt{n}))$. Ці результати були узагальнені для довільних числових полів у так звані S-Unit атаки [13], які дозволяють вирішувати проблему Ideal-SVP на експоненційний фактор швидше, проте узагальнень S-Unit атак для модульних решіток не відомо.

Взагалом, Module-LWE можливо звести за поліноміальний час до проблеми Ring-LWE за допомогою редукції, що запропонована в [4], і перейти від модульних решіток до ідеальних решіток. Проте, розмірність решітки при цьому значно зростає, що робить такий підхід непрактичним для атаки. До того ж, кількість рівнянь у Module-LWE необхідна більша, ніж є для такої редукції для існуючих криптографічних схем.

Тож, оскільки невідомо методів використання структури модуля для значного зменшення складності пошуку найменшого вектора, то надалі будемо вважати, що використовуються звичайні LWE решітки.

1.3. Алгоритми редукції решіток

Алгоритми редукції решіток на основі заданого базису решітки $(b_1,...,b_n)$ отримують інший базис, у якого ГШ-довжини відносно коротші.

Алгоритм LLL виконує над базисом дві операції: редукція за розміром $b_i \leftarrow b_i - round(\mu_{ij})b_j$ для $j \in [i-1]$ та перестановки b_i і b_{i+1} якщо $\left\|b_{i+1}^*\right\|^2 \le 1/2\left\|b_i^*\right\|^2$ поки відбуваються зміни.

У алгоритмі BKZ (та його варіаціях) фіксується розмір блоку β і відбувається пошук найменшого вектора на решітках $L_{[i,i+\beta'-1]}$ для i від 1 до $n-1$, де $\beta' = \min(\beta, n-i+1)$. Пошук вектора відбувається окремою процедурою (SVP-оракулом).

Для індекса i стандартна реалізація алгоритма BKZ викликає SVP-оракул для решітки $L_{[i,i+\beta-1]}$ і знаходить найкоротший вектор v на цій решітці. Далі BKZ вставляє v у старий базис між b_{i-1} та b_i. Для базису $(b_1,...,b_{i-1},v,b_i,...,b_{min(i+\beta-1,n)})$ застосовується LLL (або BKZ з меншим розміром блока) для отримання нового базису з меншими векторами. Ці процедури складають один раунд алгоритму. У оригінальній версії BKZ алгоритм зупинявся, коли оновлень не відбувалось на протязі $n-1$ раундів.

Наразі існує багато узагальнень BKZ. Особливо варто відмітити слайд редукцію [14], SD-BKZ [15] та G6K [16]. Ці алгоритми мають кращу асимптотичну поведінку. Проте, варто зауважити, що усі узагальнення BKZ роблять поліноміальну кількість викликів до певних алгоритмів пошуку найменшого (або малого) вектора.

Базис $(b_1,...,b_n)$ є HKZ-редукованим, якщо $|\mu_{ij}| \le 1/2$ для всіх i та j і $\pi_i(b_i)$ є найменшим вектором на проєктивній підрешітці $L_{[i:n]}$ для всіх i. ГШ-довжини при цьому можливо оцінити як $\left\|b_i^*\right\| = GH(L_{[i:n]})$.

Мірою якості редукції є кореневий фактор ерміта, який визначений наступним чином:

$$\delta = \left(\|b_0\| / vol(\Lambda)^{1/d}\right)^{1/d}. \quad (6)$$

2. Механізми інкапсуляції ключів

Відповідно до визначення [17] протокол інкапсуляції ключів є трійкою алгоритмів $(Gen, Encaps, Decaps)$, де $Gen: 1^{\lambda} \to (pk, sk)$ – поліноміальний ймовірнісний алгоритм генерації ключової пари. Приймає параметр безпеки 1^{λ} та повертає ключову пару (pk, sk); $Encaps: pk \to (K, C)$ – поліноміальний ймовірнісний алгоритм інкапсуляції ключа. Приймає публічний ключ pk і повертає випадковий ключ K та його інкапсуляцію C; $Decaps: (sk, C) \to \{K, \perp\}$ – детермінований поліноміальний алгоритм декапсуляції ключа. Приймає секретний ключ sk та інкапсуляцію ключа C і повертає ключ K у разі вдалої декапсуляції та символ помилки $\perp$ у разі виникнення помилок.

2.1. Механізм інкапсуляції ключів CRYSTALS-Kyber

Механізм інкапсуляції ключів CRYSTALS-Kyber [2] використовує перетворення у полі $R_q = Z_q[X]/(X^n + 1)$ і грунтується на проблемі Module-LWE. Елементи поля представляються у вигляді поліномів.

У CRYSTALS-Kyber використовуються наступні криптографічні геш функції:

$$PRF: \{0,1\}^{32} \times \{0,1\}^{8} \to \{0,1\}^{*}$$
$$XOF: \{0,1\}^{*} \times \{0,1\}^{8} \times \{0,1\}^{8} \to \{0,1\}^{*}$$
$$H: \{0,1\}^{*} \to \{0,1\}^{32}$$
$$G: \{0,1\}^{*} \to \{0,1\}^{32} \times \{0,1\}^{32}$$
$$KDF: \{0,1\}^{*} \to \{0,1\}^{*}$$

Додатково використовується геш-функція $Parse$, яка перетворює бітову строку на елемент поля з рівномірного розподілу (при умові, якщо вхідні дані з рівномірного розподілу).

Для генерації вектору шуму використовується біноміальний розподіл B_{η} з параметром η. Відповідно, для генерації векторів поліномів з біноміального розподілу використовується функція CBD_{η}.

Схема асиметричного шифрування, що використовується у CRYSTALS-Kyber, зображена на рис. 1.

$KyberKEM.Gen()$:	$KyberKEM.Encaps(pk)$:	$KyberKEM.Decaps(c, sk)$:
1. $z \leftarrow \{0,1\}^{32}$	1. $m \leftarrow \{0,1\}^{32}$	1. $m_1 = KyberPKE.Dec(sk, c)$
2. $(pk, sk_1) = KyberPKE.Gen$	2. $m = H(m)$	2. $(K_1, r_1) = G(m_1 \| h)$
3. $sk = (sk_1, pk, H(pk), z)$	3. $(K, r) = G(m \| H(pk))$	3. $c = KyberPKE.Enc(pk, m_1, r_1)$
4. Повернути (pk, sk)	4. $c = KyberKEM.Enc(pk, m, r)$	4. $K = \begin{cases} KDF(K_1 \| H(c)), c == c_1 \\ KDF(z \| H(c)), c \neq c_1 \end{cases}$
	5. $K = KDF(K \| H(c))$	5. Повернути K
	6. Повернути (c, K)	

Рис. 1. Асиметрична схема KyberPKE

Для отримання протокола інкапсуляції ключів використовується варіант перетворення Фуджісакі – Окамото з неявним відхиленням [18] (рис. 2).

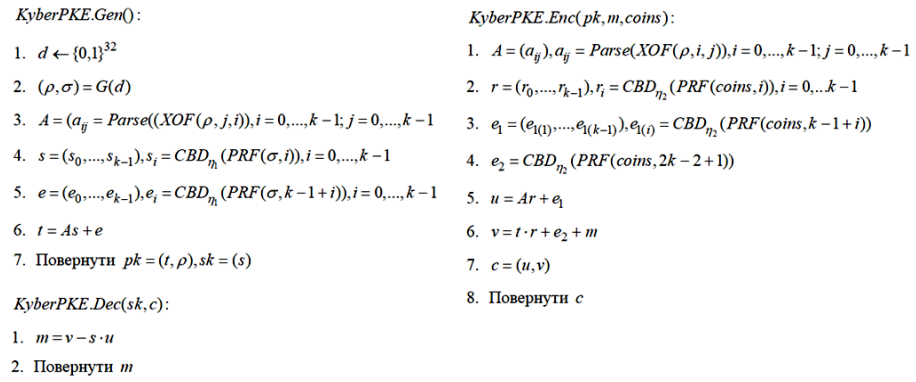

Рис. 2. Механізм інкапсуляції ключів CRYSTALS-Kyber

2.2. Механізм інкапсуляції ключів ДСТУ 8961:2019

ДСТУ 8961:2019 [1] використовує перетворення у полі $R_q = Z_q[X]/(X^n - X - 1)$ і грунтується на проблемі NTRU. Позначимо як R_3 множину усіх поліномів поля R_q, усі коефіцієнти яких належать до множини $\{-1,0,1\}$, як $R_3^{a,b}$ множину усіх поліномів у R_3, що мають кількість ненульових елементів у діапазоні $[a,b]$. Якщо $a = b$, то використовується скорочене позначення $R_3^{t,t} = R_3^t$.

ДСТУ 8961:2019 використовує наступні геш-функції:

$$BPGM : \{0,1\}^L \times R_q \to R_q$$

$$MGF : R_q \to R_3$$

$$H : R_q \to \{0,1\}^\lambda$$

$$KDF : R_q \to \{0,1\}^{K_{len}}.$$

де λ – параметр безпеки, t – загальносистемний параметр, від якого залежить кількість ненульових елементів в поліномах, L – повна довжина повідомлення, K_{len} – довжина ключа інкапсуляції. Додатково використовується бієктивне відображення

$$Pad : \{0,1\}^L \to R_3$$

$$Pad^{-1} : R_3 \to \{0,1\}^L.$$

Механізм інкапсуляції ключів ДСТУ 8961:2019 використовує CPA-to-CCA перетворення власної розробки. На рис. 3 зображено асиметричну схему шифрування, що лежить в основі стандарта і на рис. 4 зображено протокол інкапсуляції ключів ДСТУ 8961:2019.

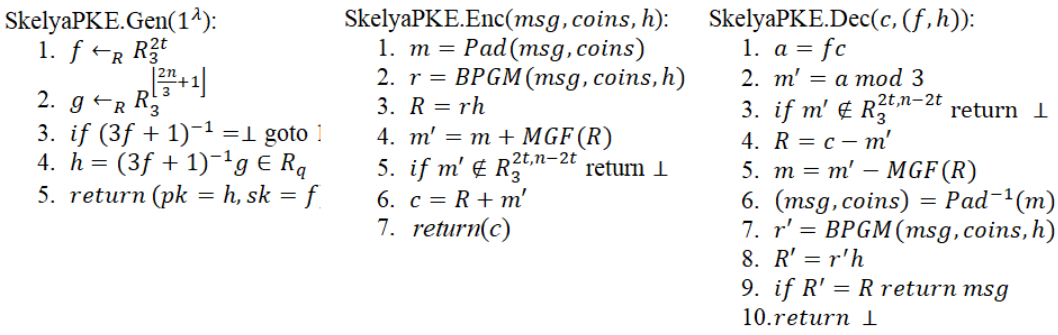

Рис. 3. Асиметрична схема SkelyaPKE

SkelyaKEM.Gen(1^{λ}):
1. $return\ (pk, sk) = SkelyaPKE.Gen(1^{\lambda})$

SkelyaKEM.Encaps($pk = h$):
1. $x \leftarrow_R \{0,1\}^{MsgLen}$
2. $r = BPGM(x, h)$
3. $C_1 = SkelyaPKE.Enc(x, r, pk)$
4. $if\ C_1 = \perp$ goto 1
5. $C_2 = H(r)$
6. $K = KDF(r)$
7. $C = (C_1, C_2)$
8. 8. $return\ (C, K)$

SkelyaKEM.Decaps($C = (C_1, C_2), sk = (f, h)$):
1. $x = SkeylaPKE.Dec(C_1, sk)$
2. $if\ x = \perp$ return $\perp$
3. $r = BPGM(x, h)$
4. $C_2' = H(r)$
5. $C_1' = SkelyaPKE.Enc(x, r, h)$
6. $if\ C_1' = C_1\ \&\&\ C_2' = C_2$
7. $return\ K = KDF(r)$
8. $return\ \perp$

Рис. 4. Механізм інкапсуляції ключів ДСТУ 8961:2019

3. Пряма атака на LWE

Розглянемо більш детально зв'язок проблем NTRU та LWE з теорією решіток.

LWE-рівняння $c - A \cdot s = e \bmod q$ можливо переписати у матричному вигляді наступним чином:

$$B \cdot \begin{pmatrix} * \\ s \end{pmatrix} + \begin{pmatrix} c \\ s \end{pmatrix} = \begin{pmatrix} e \\ s \end{pmatrix}$$
$$B = \begin{pmatrix} qI & -A \\ 0 & I \end{pmatrix} \quad (7)$$

або як

$$B \cdot \begin{pmatrix} * \\ s \\ 1 \end{pmatrix} = \begin{pmatrix} e \\ s \\ t \end{pmatrix}$$
$$B = \begin{pmatrix} qI & -A & c \\ 0 & I & 0 \\ 0 & 0 & t \end{pmatrix}, \quad (8)$$

де t – довільна константа та символ $*$ позначає будь-який вектор. З рівняння видно, що базис B задає решітку, яка містить секретний вектор s з нормою $\sqrt{(n+m)\cdot\sigma^2 + t^2}$.

Нехай $d = n + m + 1$. Якщо $\sqrt{(n+m)\cdot\sigma^2 + t^2} < GH(\Lambda(B)) \approx \sqrt{\frac{d}{2\pi e}} \cdot q^{n/d}$, то B містить вектор, що менший за очікуване значення для випадкових решіток і, отже, має розподілення векторів, що відрізняється від припущень, що використовуються при аналізі BKZ.

Пошук цільового вектора на решітці, що задається базисом B з рівняння (7), зводиться до проблеми BDD, яка формально визначена наступним чином.

Проблема α-BDD. Нехай задано базис B та вектор t і параметр $0 < \alpha < 1/2$, для яких $dist(t, B) < \alpha \cdot \lambda_1(B)$. Необхідно знайти вектор $v \in \Lambda(B)$, який є найближчим до t.

Умова $0 < \alpha < 1/2$ гарантує існування унікального рішення. При $1/2 < \alpha \le 1$ унікальне рішення існує з великою ймовірністю.

LWE з поліноміальною кількістю рівнянь може бути розглянуто як BDD (формула (7)). Асимптотично для будь-якого поліноміально обмеженого $\gamma \ge 1$ існує редукція від $BDD_{1/(\sqrt{2}\gamma)}$ до проблеми $uSVP_{\gamma}$, яка визначена наступним чином:

Проблема $uSVP_{\gamma}$. Нехай задана решітка Λ, для якої виконується $\lambda_2(\Lambda) > \gamma \cdot \lambda_1(\Lambda)$. Необхідно знайти ненульовий вектор $v \in \Lambda$ довжини $\lambda_1(\Lambda)$.

Отже, проблему LWE можливо звести до пошуку малого вектора на решітці з базисом, що задається формулою (8).

4. Пряма атака на NTRU

NTRU решітка має вигляд

$$\Lambda_H^q = \left\{(x, y) \in Z^{2n} \mid Hx - y = 0 \bmod q\right\}, \tag{9}$$

де H є матрицею, у якої i-й стовбець містить коефіцієнти $x^i \cdot h$ для деякого полінома h для $i = 0,...,n-1$. Відповідно базисом NTRU решітки є

$$B = \begin{pmatrix} qI & H \\ 0 & I \end{pmatrix}. \tag{10}$$

Найменшим вектором на решітці є (f, g). Якщо $\|f, g\|$ є меншим за $GH(\Lambda_H^q) \approx \sqrt{n/(\pi e)} \cdot \sqrt{q}$, то решітка містить незвично малий вектор і розподіл малих векторів значно відрізняється від випадкових решіток. Більш того, найменший вектор є не унікальним. Вектори $(x^i \cdot f, x^i \cdot g)$ для $i = 0,...,n-1$ також лежать на решітці.

Якщо не брати до уваги алгебраїчну структуру решітки, то проблему NTRU можливо розглядати як проблему uSVP на решітці з базисом, що задається формулою (9).

Проте, оскільки у NTRU решіток існує багато малих векторів, то це дає додаткову збитковість, яку можна використовувати, на відміну від LWE решіток. У роботах [19, 20] досліджено вплив цієї збитковості. Якщо $q >> n$, то у багатьох випадках можливо пришвидшити редукцію на експоненційний фактор. Параметри криптографічних схем на решітках зазвичай не задовольняють цій умові, проте доволі важко визначити межу, коли прискорення можливо отримати, а коли ні.

Тож, проблеми LWE та NTRU можна звести до проблеми uSVP на решітках з базисами (8) та (10) відповідно.

5. Модель coreSVP

У загальному випадку доволі важко визначити конкретні оцінки часу редукції решітки. Це пов'язано з тим, що асимптотичні оцінки алгоритмів редукції сильно відрізняються від експериментальних даних. Зазвичай алгоритм BKZ та його узагальнення дозволяють отримати менші вектори, ніж зазначено в асимптотичних оцінках.

У роботі [6] запропонована методологія, яка є стандартною методологією для усієї криптографії на решітках. Запропонований підхід дозволяє отримати нижню оцінку часу роботи. Оскільки BKZ та інші алгоритми редукції решіток роблять поліноміальну кількість викликів до SVP-оракула, то автори методології запропонували оцінити складність атаки як час роботи SVP-оракула на решітці розмірності β, де β – найменший розмір блоку для алгоритму BKZ, який дозволяє отримати базис, що містить цільовий вектор.

Визначення мінімальної необхідної розмірності β також є нетривіальною задачею і вирішується за допомогою евристик. У межах цієї роботи використовується еврістичний підхід, що був запропонований в роботі [6]. Сутність евристики полягає у наступному припущенні. Нехай d – розмірність решітки Λ. Якщо під час останнього виклику SVP-оракула для решітки $\Lambda_{[d-\beta:d]}$ буде знайдений вектор $\pi_{d-\beta}(v)$, то з високою ймовірнісю у безисі решітки Λ буде міститися цільовий вектор v після завершення раунду BKZ. Відповідно для мінімальної розмірності β має виконуватися нерівність

$$\left\|\pi_{d-\beta}(v)\right\| < \left\|b^*_{d-\beta}\right\|. \tag{11}$$

У роботі [21] проведено експериментальний аналіз цього підходу для LWE решіток і показано, що така умова вдалого завершення атаки є доволі точною. Тож, проблема зводиться до оцінки значень $\left\|b^*_{d-\beta}\right\|$.

Оскільки теоретичні оцінки доволі сильно відрізняються від експериментальних даних, то використовується ряд евристик. Розглянемо як змінюються логарифми від значень ГШ-норм $\left\|b^*_i\right\|$. На рис. 5 графік значень $\ln\left\|b^*_i\right\|$ для LWE решітки (зліва) та NTRU решітки (праворуч) до та після BKZ редукції з розміром блоку 60.

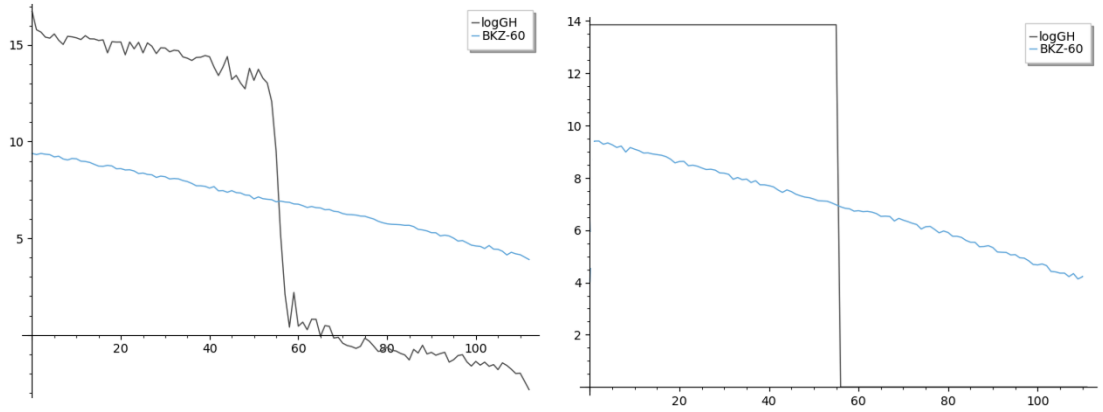

Рис. 5. Графіки ГШ-норм для LWE решітки та NTRU решітки
до та після BKZ редукції з розміром блоку 60

З рис. 5 видно, що після редукції графік ГШ-норм має вигляд прямої. З визначення кореневого фактора ерміта (6) випливає, що $\left\|b^*_0\right\| = \|b_0\| = \delta^d \det(\Lambda)^{1/d}$. Евристика GSA (Geometric Series Assumption) стверджує, що ГШ-норми утворюють геометричну прогресію з параметром δ^2.

Точне значення параметра δ для BKZ не відоме, проте доволі точним наближенням на основі евристики Гауса вважається

$$\delta = \left((\pi\beta)^{1/\beta} \cdot \beta / 2\pi e\right)^{1/2(\beta-1)}. \tag{12}$$

Формула (11) зазвичай використовується при аналізі атак, проте у роботі [7] була запропонована точніша асимптотична оцінка:

$$\delta = \left((\pi\beta)^{1/\beta} \cdot \beta / 2\pi e\right)^{1/2(\beta-1)+\beta/(2n^2)}. \tag{13}$$

Відповідно, перебираючи значення β, можливо знайти найменше, для якого досягається достатньо мале значення δ, щоб (12) або (13) виконувалася.

У табл. 1 – 2 наведено результати експериментальної перевірки оцінок (12) та (13) на решітках малої розмірності для значень $\beta = 20, 50$.

Таблиця 1

Перевірка оцінок (11) та (12) для $\beta = 20$

n	$\log_2 q$	$\log_2 \|b_0\|$	Середньоквадратична помилка для (11)	Середньоквадратична помилка для (12)
50	20	10.50	0.17	0.28
	40	20.34	0.33	0.44
70	20	10.91	0.03	0.11
	40	20.98	0.04	0.04
90	20	11.69	0.48	0.42
	40	21.56	0.35	0.29
110	20	12.10	0.62	0.57
	40	22.21	0.73	0.68
130	20	12.72	0.91	0.92
	40	22.66	0.91	0.86

З табл. 1 – 2 видно, що для розміру блоку 20 оцінка (13) спочатку є більше значення середньоквадратичної помилки, проте з ростом розмірності решітки досягає менших значень помилки, ніж у (12). Проте, для розміру блоку 50 на малих розмірностях не вдалося досягти менших помилок, ніж для (11). З табл. 1 – 2 випливає, що можливо оцінити значення $\|b_0\|$ з точністю до деякого поліноміального фактору.

Таблиця 2

Перевірка оцінок (11) та (12) для $\beta = 50$

n	$\log_2 q$	$\log_2 \|b_0\|$	Середньоквадратична помилка для (11)	Середньоквадратична помилка для (12)
50	20	10.86	0.71	1.55
	40	20.34	0.52	1.36
70	20	10.67	0.53	1.13
	40	20.69	0.51	1.11
90	20	11.12	0.42	0.89
	40	21.07	0.47	0.94
110	20	11.13	0.75	1.14
	40	21.17	0.71	1.10
130	20	11.61	0.62	0.94
	40	22.12	0.63	0.92

На рис. 6 зображено евристику GSA для LWE та NTRU решіток на основі формули (11).

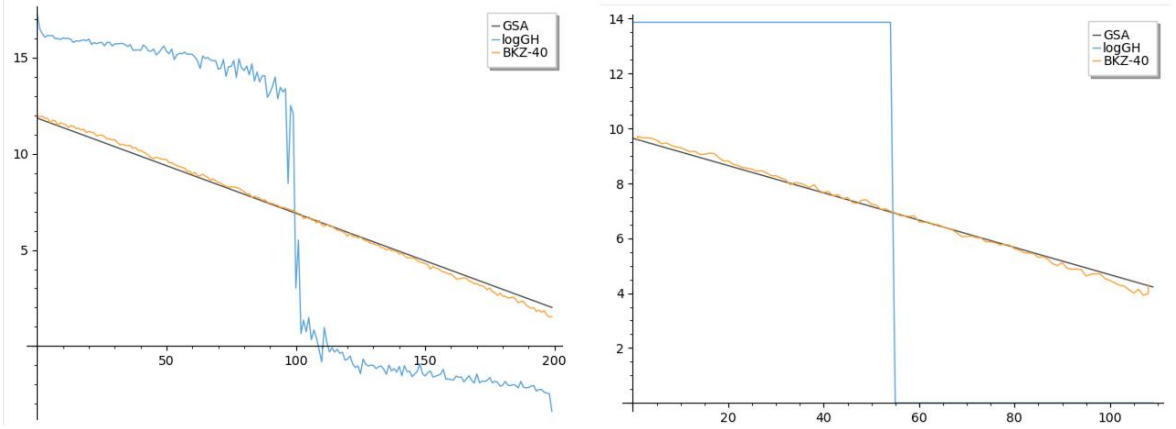

Рис. 6. Евристика GSA

З рис. 6 видно, що евристика GSA добре працює при великих відносно розмірності решітки блоках, проте, якщо розмір блоку є значно меншим за розмірність, то GSA буде виконуватися не для всіх векторів у решітці. Приклад такого явища для LWE та NTRU решіток зображено на рис. 7.

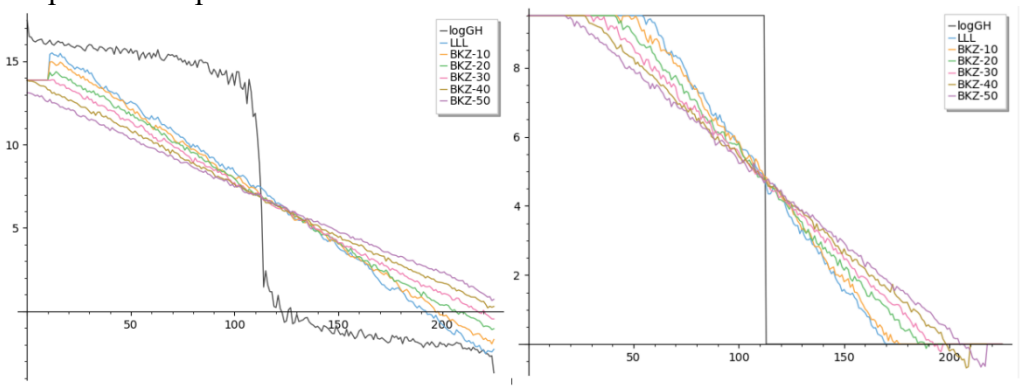

Рис. 7. Приклад ГШ-норм, що частково порушують евристику GSA

Таке явище має місце, оскільки детермінант решітки $\det(\Lambda)=\prod_i \left\|b_i^*\right\|$ є інваріантом. Форми прямої для деяких факторів δ неможливо досягти, оскільки не існує таких векторів, щоб детермінант зберігався. Деяке число перших ГШ-векторів буде мати значення норми q (оскільки LWE решітки та NTRU решітки є q-арними решітками, то вони містять вектори вигляду $(0,...,q,...,0)$ і ці вектори є відносно малими), а останні вектори будуть мати значення норми близьке до 1, оскільки значення детермінанту повинно бути інваріантом. Оцінити кількість векторів з значенням норми q доволі легко. Обчислювати значення вектора згідно з GSA і заміняти його на q, якщо воно є більшим. Такий підхід є евристикою ZGSA. На рис. 8 наведено приклад евристики ZGSA. З рис. 8 видно, що для NTRU решіток ZGSA добре апроксимує значення логарифмів ГШ-норм. Для LWE решіток має місце артефакт. Частина значень, що повинна була мати значення q насправді має значення, відповідають евристиці GSA. Це пов'язано з малою кількістю раундів BKZ (8 раундів). З ростом розмірності і кількості раундів BKZ евристика ZGSA даватиме точніший результат.

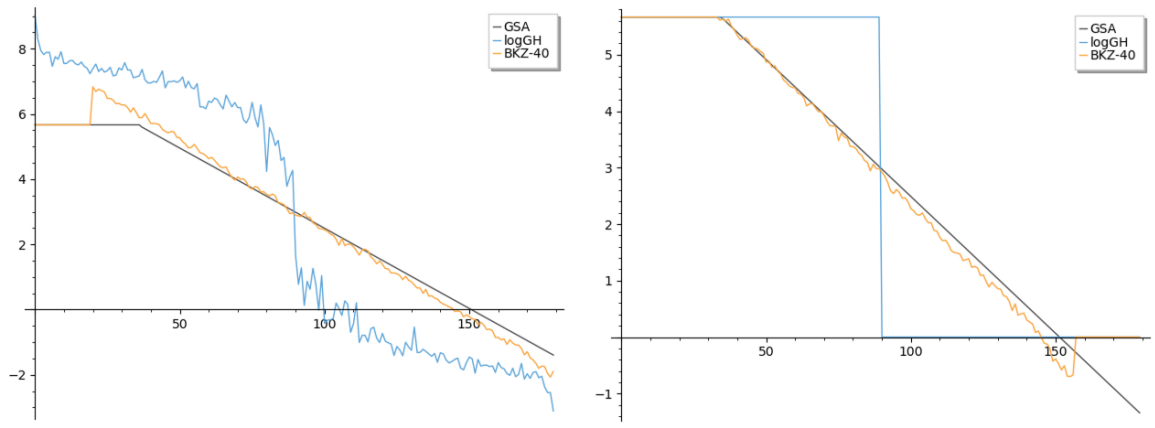

Рис. 8. Евристика ZGSA

Використання ZGSA є стандартним підходом у криптографії на решітках. До недоліків такого підходу можливо віднести те, що важко оцінити наскільки точною буде ця евристика на дійсно великих розмірностях, що відповідають криптографічним значенням параметрів.

Використання ZGSA і евристики (10) дозволяє визначити мінімальне значення β, за якого атака буде успішною. Останнім компонентом для побудови моделі є визначення часу

роботи SVP оракула в залежності від значення β. Двома найпоширенішими підходами до побудови SVP-оракулів є алгоритми просіювання (sieving algorithms) та алгоритми переліку точок (enumeration algorithms) [22].

Алгоритми просіювання грунтуються на суто геометричних ідеях. На початку алгоритму генерується субекспоненційна кількість векторів на решітці. Далі обчислюється множина векторів, що є різницею згенерованих векторів (відбувається "просіювання"). У наступний раунд потрапляють тільки ті вектори, що мають менші довжини, ніж у згенерованих. Процес "просіювання" повторюється ітеративно, поки "просіювання" повертає не пусту множину. Якщо на початку була достатня кількість векторів, то через поліноміальну кількість ітерацій буде знайдено близький до найменшого вектор. Найкращий алгоритм [23] цього класу на сьогоднішній день потребує $\sqrt{3/2}^{\beta/2+o(\beta)}$ операцій. До табл. 3 зведено оцінки роботи найкращих алгоритмів просіювання, згідно з [22].

Зауважимо, що фактор $o(\beta)$ під час аналізу ігнорується, проте він може бути доволі великим. Реальна вартість атаки може коштувати значно більше, тому у табл. 3 зведено як теоретичну оцінку, так і оцінку, що отримана на основі експериментальних даних [22]. Квантові комп'ютери дають відносно невелике прискорення алгоритмів просіювання. Це пов'язано з тим, що операцію «просіювання» важко прискорити за допомогою квантових ефектів.

Таблиця 3

Оцінки часу роботи алгоритмів просіювання

Оцінка	Формула
Алгоритм просіювання на класичному комп'ютері (теоретична оцінка)	$2^{0.29248125036\beta+o(\beta)}$
Алгоритм просіювання на класичному комп'ютері (експерементальна оцінка)	$2^{0.3924062518\beta-5}$
Алгоритм просіювання на квантовому комп'ютері	$2^{0.265\beta+o(\beta)}$

Алгоритми переліку точок є комбінаторним методом вирішення задачі SVP на певній решітці. Для заданного базису $(b_1,...,b_n)$ та ГШ-базису $(b_1^*,...,b_n^*)$ алгоритм переліку полягає у побудові дерева пошуку, у якому вузлами є вектори. Корінь дерева є нульовим вектором. Для кожного вузла $v \in L$ на глибині $k \in 1,...,n$ потомки містять вектори $v+\alpha_{n-k}b_{n-k}(\alpha_{n-k} \in Z)$ з довжиною $\left\|\pi_{n-k}\left(\sum_{i=n-k}^{n} a_i b_i\right)\right\|$ меншою за задану константу $R_{k+1} \in \left(0, \|b_1\|\right]$. Після проходу по всім можливим вузлам алгоритм знаходить вектор на решітці, що менший за радіус пошуку R_n на певній глибині.

Щоб зменшити радіус пошуку, доцільно обирати R_n найближче до норми шуканого вектора. Відповідно, для $\lambda_1(L)$ у якості константи можливо задати $R_n = GH(L)$, тоді кількість векторів, відповідно до евристики Гауса, яку необхідно перебрати, становить

$$FEC(B) = \sum_{k=1}^{n} \frac{V_k(GH(L))}{\prod_{i=n-k+1}^{n} \left\|b_i^*\right\|} \ . \tag{14}$$

Щоб зменшити необхідну кількість векторів, різними авторами були запропоновані евристичні покращення. У роботі [22] запропоновано модель оцінки вартості алгоритмів переліку, у межах якої були запропоновані обмеження на $R_1,...,R_n$. При цьому, на відміну від оригінального алгоритму, така оптимізація робить знаходження необхідного вектора ймовірнісною подією.

Ймовірність p знаходження вектора у шарі з радіусом c, згідно з визначенням, становить

$$p = \Pr_{X \leftarrow S_n}\left[\sum_{i=1}^{l} x_l^2 < R_l^2, l \in 1,...,n\right]. \quad (15)$$

При цьому ціна (кількість векторів) пошуку малого вектора складатиме

$$N = \frac{1}{2}\sum_{k=1}^{n} \frac{vol\{(x_1,...,x_k \in R^k : \sum_{i=1}^{l} x_i^2 < R_l^2, l \in 1,...k\}}{\prod_{i=n-k+1}^{n} \left\| b_i^* \right\|}. \quad (16)$$

З використанням цієї методології у роботі [24] знайдені оптимальні обмеження на $R_1,...,R_n$. У табл. 4 зведено оцінки сучасних алгоритмів переліку з врахуванням описаних вище оптимізацій.

Квантові комп'ютери здатні значно краще прискорити алгоритми переліку, порівняно з алгоритмами просіювання. Прискорення отримується за допомогою застосування варіантів алгоритма Гровера. Алгоритми переліку потребують значно менше пам'яті. Можливість їх реалізації на квантових комп'ютерах ймовірно з'явиться значно раніше, ніж для алгоритмів просіювання, проте, з табл. 4 видно, що асимптотично вони працюють гірше.

Таблиця 4

Оцінки часу роботи алгоритмів переліку

Оцінка	Формула
Класичний комп'ютер, робота [23]	$2^{0.125\beta \cdot \log_2(\beta) - 0.547\beta + 10.4}$
Класичний комп'ютер, робота [34]	$2^{0.1839\beta \cdot \log_2(\beta) - 0.995\beta + 16.25}$
Квантовий комп'ютер	$2^{(0.125\beta \cdot \log_2(\beta) - 0.547\beta + 10.4)/2}$ $2^{(0.1839\beta \cdot \log_2(\beta) - 0.995\beta + 16.25)/2}$

6. Оцінка на основі GSA

Оцінки для CRYSTALS-Kyber у моделі core-SVP для алгоритмів на основі просіювання наведені у табл. 5, з якої видна значна різниця між оцінками безпеки CRYSTALS-Kyber з урахуванням експериментальних результатів та без. Хоча алгоритми просіювання є асимптотично швидшими, проте затрати на пам'ять можуть значно зменшити їх ефективність.

Таблиця 5

Оцінки безпеки CRYSTALS-Kyber
у моделі core-SVP для алгоритмів просіювання

Модель	Класичний комп'ютер (теоретична)	Класичний комп'ютер (експериментальна)	Квантовий комп'ютер
Kyber512	118	155	107
Kyber768	183	243	166
Kyber1024	256	342	232

В табл. 6 наведено оцінки безпеки CRYSTALS-Kyber у моделі core-SVP для алгоритмів переліку. З таблиці видно, що вартість алгоритму переліку зростає асимптотично швидше, проте на квантових комп'ютерах вони мають більше прискорення у порівнянні з алгоритма-

ми просіювання. З урахуванням того, що вони потребують менше пам'яті, можливо для набору параметрів Kyber512 має сенс використовувати саме алгоритми переліку на квантових комп'ютерах.

Таблиця 6

Оцінки безпеки CRYSTALS-Kyber
у моделі core-SVP для алгоритмів переліку

Модель	Класичний комп'ютер	Квантовий комп'ютер
Kyber512	228	114
Kyber768	394	197
Kyber1024	603	301

Оцінки для ДСТУ 8961:2019 у моделі core-SVP для алгоритмів просіювання наведені у табл. 7, якої видно, що для квантових комп'ютерів ДСТУ 8961:2019 забезпечує необхідний рівень безпеки, проте для класичних комп'ютерів оцінки є дещо нижчими за необхідний рівень безпеки.

Таблиця 7

Оцінки безпеки ДСТУ 8961:2019 у моделі core-SVP
для алгоритмів просіювання

Модель	Класичний комп'ютер (теоретична)	Класичний комп'ютер (експериментальна)	Квантовий комп'ютер
Скеля256	169	224	153
Скеля384	239	319	217
Скеля512	296	397	268

Оцінки для ДСТУ 8961:2019 у моделі core-SVP для алгоритмів переліку наведені у табл. 8. Для алгоритмів переліку ДСТУ 8961:2019 забезпечує заявлені рівні безпеки.

Таблиця 8

Оцінки безпеки ДСТУ 8961:2019 у моделі core-SVP
для алгоритмів переліку

Модель	Класичний комп'ютер	Квантовий комп'ютер
Скеля256	358	179
Скеля384	554	277
Скеля512	722	361

7. Оцінка на основі симулятора Чена – Нгуєна

Евристика ZGSA є доволі спрощеною оцінкою форми ГШ-норм. Більш сучасним підходом є оцінки на основі симуляції. Вперше підхід на основі симуляції був запропонований у роботі [8] (симулятор Чена – Нгуєна).

Симулятор працює наступним чином. Нехай $(l_1,...,l_n)$ є симульовані значення ГШ-довжин $\left\|b_i^*\right\|$ для $i=1,...,n$. Тоді симульовані значення для детермінанта і евристики Гауса будуть $\prod_{j=1}^{n} l_i$ та $GH(L_{[i:i+\beta-1]}) = V_{\beta'}(1)^{-1/\beta'} \prod_{j=i}^{i+\beta'-1} l_i$, де $\beta' = \min\{\beta, n-i+1\}$.

Для симуляції раунда BKZ з розміром блоку β передбачається, що кожен виклик алгоритму переліку знаходить вектор з довжиною $GH(L_{[i:i+\beta-1]})$. Далі знайдений вектор використовується для оновлення поточного блоку. Значення (l_i, l_{i+1}) оновлюються до (l_i', l_{i+1}') для $i = 1, \ldots, n-1$, де $l_i' = GH(L_{[i:i+\beta-1]})$ та $l_{i+1}' = l_{i+1} \cdot (l_i / l_i')$.

Останні 45 ГШ-довжин модифікуються за допомогою ГШ-довжин HKZ-редукованого базису, який обчислений усередненням експериментально отриманих редукованих базисів.

На рис. 9 зображено порівняння евристики GSA та симулятора Чена – Нгуєна. З рисунку видно, що симулятор Чена – Нгуєна може набагато точніше передбачувати форму ГШ-норм.

Проте, недоліком підходу на основі симуляції є те, що він не враховує структурованість решіток і іноді не може точно врахувати ефекти, що виникають при малих розмірах блоку (відносно розмірності решітки).

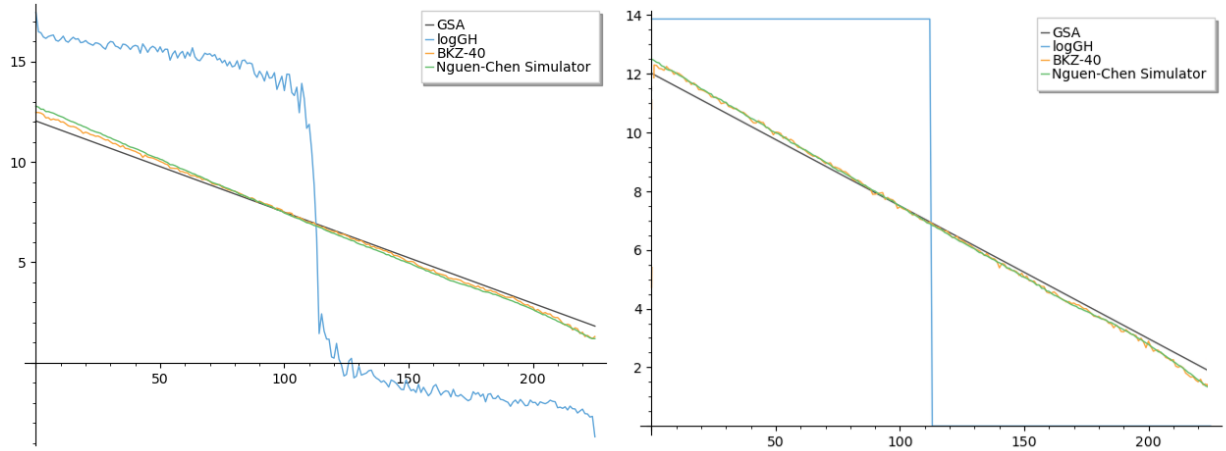

Рис. 9. Порівняння GSA та симулятора Чена – Нгуєна

На рис. 10 наведений приклад такої ситуації для LWE решітки. Симулятор дає на виході GSA-подібний графік, у той час, як перші та останні ГШ-норми не підпорядковуються GSA в результаті редукції.

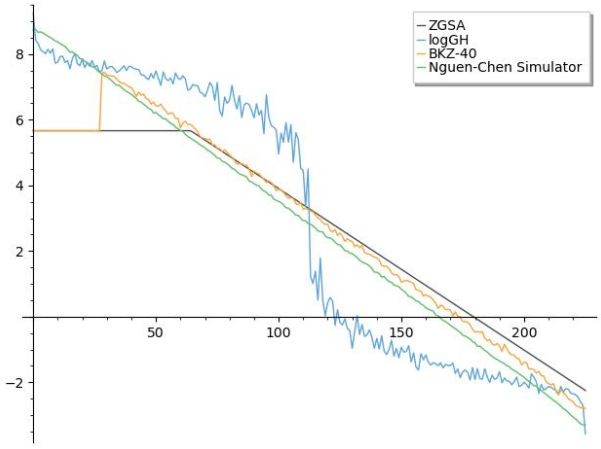

Рис. 10. Приклад неточності симуляції для LWE решітки

Іноді можливі ситуації, коли ані GSA, ані ZGSA, ані симуляція не дають достатньо точних результатів. Приклад такої ситуації наведений на рис. 11.

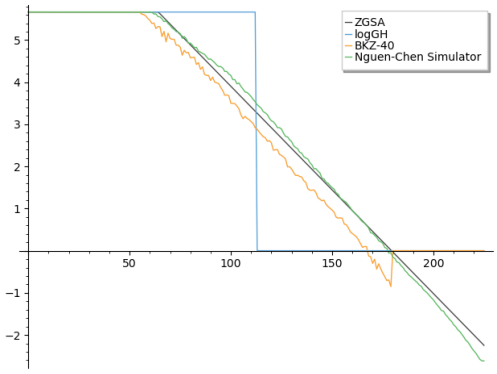

Рис. 11. Неточність евристики ZGSA та симулятора Чена – Нгуєна

Проте, більшість існуючих неточностей стосуються не криптографічних випадків. Для криптографічних випадків $\beta \approx 0.4d$, де d – розмірність решітки. За тих параметрів, що використовуються в криптографії, графік ГШ-норм буде близький до GSA, і у цьому випадку симулятори дуже гарно себе показують.

В табл. 9 наведено оцінки безпеки для CRYSTALS-Kyber на основі симулятора Чена – Нгуєна для алгоритмів просіювання.

Таблиця 9

Оцінки безпеки на основі симулятора Чена – Нгуєна
для алгоритмів просіювання

Модель	Класичний комп'ютер	Класичний комп'ютер (з пам'яттю)	Квантовий комп'ютер
Kyber512	115	151	105
Kyber768	166	220	151
Kyber1024	223	297	202

В табл. 10 наведено оцінки безпеки для CRYSTALS-Kyber на основі симулятора Чена – Нгуєна для алгоритмів переліку.

Таблиця 10

Оцінки безпеки на основі симулятора Чена – Нгуєна
для алгоритмів переліку

Модельг	Класичний комп'ютер	Квантовий комп'ютер
Kyber512	220	110
Kyber768	350	175
Kyber1024	507	253

З таблиць видно, що оцінка безпеки є дещо меншою, що є наслідком врахування симулятором ефектів редукції решіток, які GSA ігнорує.

Оцінки для ДСТУ 8961:2019 у моделі core-SVP для алгоритмів просіювання наведено у табл. 11.

Таблиця 11

Оцінки безпеки ДСТУ 8961:2019 у моделі core-SVP
для алгоритмів просіювання

Модель	Класичний комп'ютер (теоретична)	Класичний комп'ютер (експериментальна)	Квантовий комп'ютер
Скеля256	163	216	148
Скеля384	228	304	206
Скеля512	280	375	254

Аналогічно, оцінки є дещо меншими, ніж при використанні GSA. У табл. 12 зведено оцінки безпеки ДСТУ 8961:2019 для алгоритмів переліку.

Таблиця 12

Оцінки безпеки ДСТУ 8961:2019 у моделі core-SVP
для алгоритмів переліку

Модель	Класичний комп'ютер	Квантовий комп'ютер
Скеля256	343	171
Скеля384	520	260
Скеля512	674	337

На рис. 12, 13 наведено діаграму отриманих оцінок безпеки для алгоритмів просіювання та алгоритмів переліку відповідно.

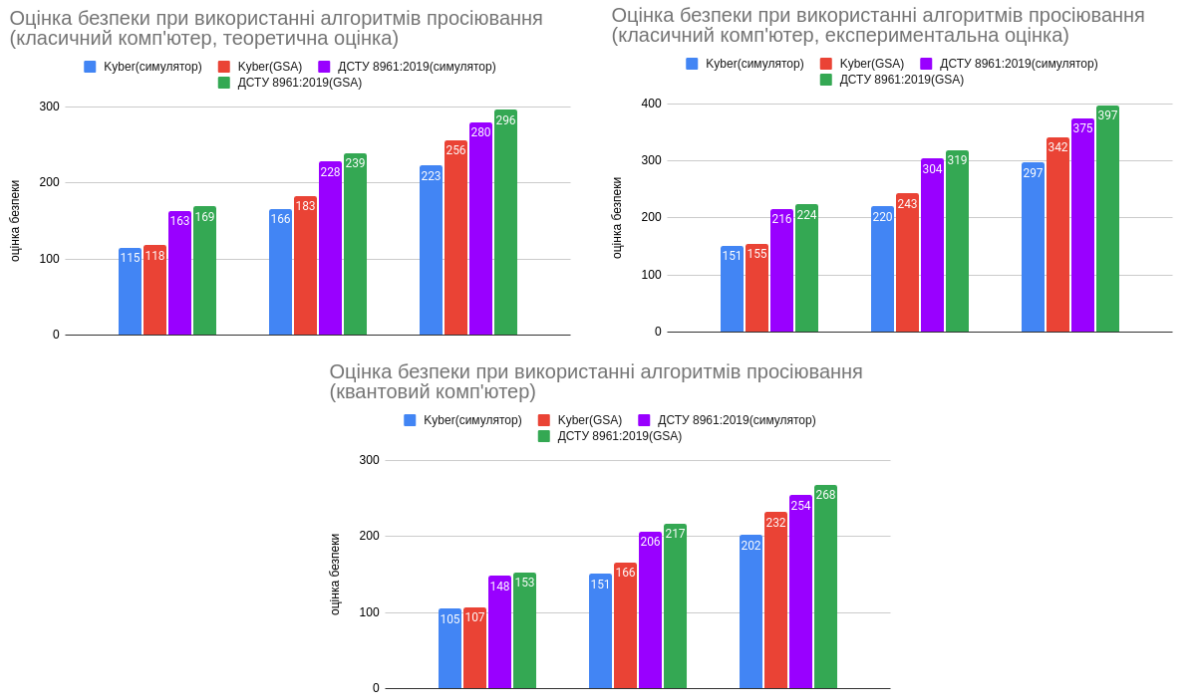

Рис. 12. Безпека CRYSTALS-Kyber та ДСТУ8961:2019 при використанні алгоритмів просіювання

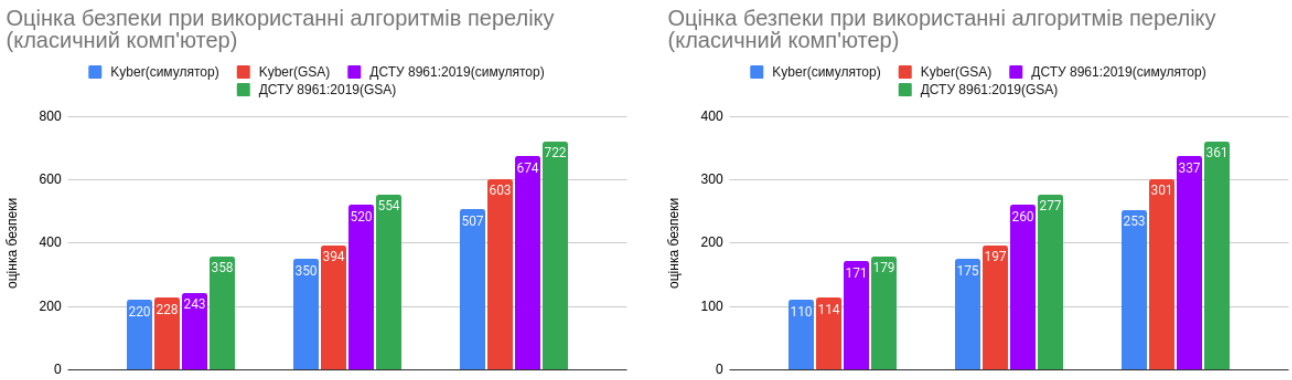

Рис. 13. Безпека CRYSTALS-Kyber та ДСТУ8961:2019 при використанні алгоритмів переліку

Висновки

1. У моделі coreSVP на практиці використовується ряд евристик. Умова вдалого завершіення атаки (нерівність (10)) є евристикою і оцінка ГШ-норм відбувається за допомогою евристики GSA (або ZGSA). Головне питання полягає в тому, наскільки ці евристики точні для решіток високої розмірності. Згідно з результатами роботи [25] умова вдалого завершення атаки дуже гарно апроксимує найменше значення β для LWE решіток. Проте, у тій же роботі автори зазначають, що при певних значеннях параметрів (які не є криптографічними) можливі випадки, коли мінімальне значення β є меншим, ніж передбачає умова вдалого завершення. У той же час, для NTRU решіток (і у деякій мірі для LWE решіток також) до кінця незрозуміло як саме впливає на мінімальне значення β велика кількість малих векторів. У роботі [20] експерментально показано, що вплив у загальному випадку є. Проте, з їх аналізу випливає, що цей ефект є релевантним для більших значень q, ніж використовуються у механізмах інкапсуляції ключів. В цілому, тема встановлення точної умови вдалого завершення атаки є відкритою, але, скоріш за все, потенційні ефекти від уточнення умови вдалого завершення атаки будуть доволі обмеженими.

2. З отриманих оцінок видно, що використання симуляторів дає менші оцінки безпеки, ніж GSA. Перші ГШ-норми та останні ГШ-норми дещо відхиляються від послідовності GSA, і з урахуванням цих ефектів оцінки безпеки є на $\approx 10-20$ біт меншими. Головне питання полягає у тому наскільки точно симулятори дозволяють оцінити норму $b^*_{d-\beta}$. З однієї сторони, на малих розмірностях легко отримати приклади неточних оцінок за допомогою симуляторів. Проте, з іншої – ці неточности пов'язані, як правило, з першими або останніми векторами і для аналізу неважливі. Автори рандомізованого симулятора Чена – Нгуєна наводять приклади ситуацій [26], коли симулятор дає надто оптимістичні оцінки. Тож, існує ймовірність, що менші значення безпеки є результатом саме такого ефекту. Використання симуляторів є перспективним напрямком, проте, необхідні більш точні симулятори, що враховують структурованість LWE та NTRU решіток.

3. Отримані оцінки безпеки показують, що для CRYSTALS-Kyber та ДСТУ8961:2019 загальносистемні параметри для квантових комп'ютерів забезпечують необхідний рівень безпеки, проте для класичних комп'ютерів для деяких наборів параметрів оцінки є нижчими за необхідний рівень.

Список літератури:

1. ДСТУ 8961:2019. Інформаційні технології. Криптографічний захист інформації. Алгоритми асиметричного шифрування та інкапсуляції ключів. Чинний від 21.12.2019. Вид. офіц. Київ : УкрНДНЦ, 2019. 72 с.

2. CRYSTALS – Kyber: a CCA-secure module-lattice-based KEM // Cryptology ePrint Archive, Report 2017/634. [Electronic resource]. Online: https://eprint.iacr.org/2017/634.pdf

3. Lyubachevsky V., Ducas L., Kiltz E. CRYSTALS-Kyber Techn. rep. NIST, 2017. [Electronic resource]. Access mode: https://pq-crystals.org/kyber/ (дата звернення: 21.03.2023)

4. Albrecht M., Deo A. Large Modulus Ring-LWE ≥ Module-LWE // URL: https://eprint.iacr.org/2017/612.pdf (дата звернення: 21.03.2023)

5. Hoffstein J., Pipher J., Silverman H. NTRU: a ring based public key cryptosystem // Algorithmic Nuber Theory. Third International Symposium. 1998. P. 267 – 288.

6. Alkim E., Ducas L., Pöppelmann T., Schwabe P. Post-quantum key exchange – a new hope // URL: https://eprint.iacr.org/2015/1092.pdf (дата звернення: 21.03.2023)

7. Li J., Nguyen P. A Complete Analysis of the BKZ Lattice Reduction Algorithm // URL: https://eprint.iacr.org/2020/1237 (дата звернення: 21.03.2023)

8. Chen Y., Nguyen P. BKZ 2.0: Better Lattice Security Estimates // ASIACRYPT, 2011.

9. Lyubashevsky V., Peikert C., Regev O. On ideal lattices and learning with errors over rings // EUROCRYPT, 2010. P. 1 – 23.

10. Eisenträger K., Hallgren S., Kitaev A., Song F. A quantum algorithm for computing the unit group of an arbitrary degree number field // Proceedings of the forty-sixth annual ACM symposium on Theory of computing. P 293 – 302. ACM, 2014.

11. Campbell P., Groves M., Shepherd D. Soliloquy: A cautionary tale. // ETSI 2nd Quantum-Safe Crypto Workshop. P. 1 – 9.

12. Biasse J., Song F. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields // ACM-SIAM symposium on Discrete Algorithms, 2017. P. 893 – 902.

13. Bernstein D., Lange T. Non-randomness of S-unit lattices // [Electronic resource]. Online: https://s-unit.attacks.cr.yp.to/spherical.html

14. Gamma N., Nguyen P. Finding short lattice vectors within Mordell's inequality // STOC, 2008. P. 3 – 13.

15. Micciancio D., Walter M. Practical, predictable lattice basis reduction // EUROCRYPT, 2016. P. 56 – 73.

16. Albrecht M., Ducas L., Herold G., Kirshanova E., Postlethwaite E., Stevens M. The General Sieve Kernel and New Records in Lattice Reduction. // URL: https://eprint.iacr.org/2019/089 (дата звернення: 21.03.2023)

17. Dent A. A Designer's Guide to KEMs. Cryptography and Coding // Cryptography and Coding, 2003. Vol 28. P. 29 – 56.

18. Hofheinz D., Hovelmanns K., Kiltz E. A modular analysis of the fujisaki-okamoto transformation // Lecture Notes in Computer Science. 2017. Vol. 10677. P. 341 – 371.

19. Cheon J., Jeong J., Lee C. An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low-level encoding of zero // LMS Journal of Computation and Mathematics, 2016. Vol. 19 P. 255 – 266.

20. Kirchner P., Fouque P. Revisiting lattice attacks on overstretched NTRU parameters // STOC, 2017. P. 3 – 26.

21. Albrecht M., Göpfert F., Virdia F., Wunderer T. Revisiting the expected cost of solving uSVP and applications to LWE // ASIACRYPT. 2017. Vol. 10624. P.297 – 322.

22. Bernstein D., Chuengsatiansup C., Lange T., Vredendaal C. NTRU Prime: reducing attack surface at low cost // URL: https://eprint.iacr.org/2016/461 (дата звернення: 21.03.2023)

23. Laarhoven T., Mariano A. Progressive lattice sieving // URL: https://eprint.iacr.org/2018/079.pdf (дата звернення: 21.03.2023)

24. Gama N., Nguyen P., Regev O. Lattice Enumeration Using Extreme Pruning // URL: https://hal.science/hal-01083526/document (дата звернення: 21.03.2023)

25. Albrecht M., Göpfert F., Virdia F., Wunderer T. Revisiting the expected cost of solving uSVP and applications to LWE // ASIACRYPT. 2017. Vol. 10624. P.297 – 322.

26. Bai S., Stehle D., Wen W. Measuring, simulating and exploiting the head concavity phenomenon in BKZ // ASIACRYPT, 2018. P. 389 – 404.

Надійшла до редколегії 15.02.2023

Відомості про авторів:

Кандій Сергій Олегович – Харківський національний університет імені В. Н. Каразіна, аспірант кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; АТ «Інститут Інформаційних Технологій», технік-конструктор, Україна; e-mail: sergeykandy@gmail.com