УДК 004.056.5 **DOI:10.30837/rt.2023.1.212.02**

Я.А. ДЕРЕВ'ЯНКО, І.Д. ГОРБЕНКО, д-р техн. наук

АТАКИ БІЧНИМИ КАНАЛАМИ НА CRYSTALS-KYBER, КОНТРЗАХОДИ ТА ПОРІВНЯННЯ З АЛГОРИТОМ СКЕЛЯ (ДСТУ 8961-2019)

Вступ

Kyber [1] – це постквантовий алгоритм на основі решітки, заснований на складності задачі M-LWE. Kyber пропонує безпечну схему шифрування з відкритим ключем (PKE) проти атаки вибраного відкритого тексту (CPA) і захищений механізм інкапсуляції ключів проти атаки вибраного шифротексту (CCA). Kyber має три рівні безпеки: Kyber512, Kyber768 і Kyber1024, де кількість постквантових бітів безпеки становлять 107, 166 і 232 відповідно.

У даній роботі проведено систематичне дослідження атак бічними каналами (SCA) і атак із впровадженням помилок (помилками) (FIA) на структуровані схеми на основі решітки, з основним акцентом на механізмі інкапсуляції ключів Kyber (KEM), який є одним із провідних кандидатів в процесі стандартизації NIST для постквантової криптографії (PQC).

Враховуючи широкий спектр відомих атак, одночасний захист від усіх атак потребує впровадження індивідуальних засобів захисту/протидії. У роботі представлено ряд індивідуальних контрзаходів, здатних забезпечити захист/пом'якшення щодо існуючих SCA/FIA для Kyber KEM [1].

Проведена на платформі на основі ARM Cortex-M4 оцінка продуктивності показує, що представлені спеціальні контрзаходи спричиняють розумні витрати на продуктивність. Тому, можна сказати, що у роботі наводено аргументи на користь використання спеціальних контрзаходів у реальних реалізаціях схем на основі решітки, або окремо, або як підсилення загальних контрзаходів, таких як маскування.

1. Атаки бічними каналами на Kyber KEM

Атаки бічними каналами на Kyber KEM можна класифікувати наступним чином: на основі мети зловмисника та на основі доступу до входів/виходів цілі.

На основі мети.

Такі атаки можна умовно розділити на дві категорії: атаки відновлення повідомлень і атаки відновлення ключа [2].

Атаки відновлення повідомлення намагаються відновити повідомлення m з дійсного зашифрованого тексту ct, що відповідає дійсному обміну ключами між двома легітимними сторонами. Знання m призводить до відновлення спільного секрету або ключа сеансу K, тим самим ставлячи під загрозу чи компрометуючи конфіденційність відповідного цільового сеансу.

Атаки на відновлення ключа намагаються відновити довгостроковий приватний ключ sk. Відновлення sk призводить до відновлення всіх сеансових ключів K, отриманих з його допомогою. В умовах, коли sk оновлюється для кожного обміну ключами, ефективність атак відновлення ключа та атак відновлення повідомлень є однаковою.

На основі доступу до входів/виходів цілі.

Такі атаки також можна поділити на дві категорії: атаки з відомим шифротекстом (KCA) і атаки з обраним шифротекстом (CCA).

Атаки з відомим шифротекстом – це атаки, які припускають, що зловмиснику відомі лише зашифровані тексти. У такому випадку зловмисник може лише пасивно спостерігати (проводити моніторинг) за цільовим пристроєм, не маючи можливості встановити з ним зв'язок.

Атаки з обраним шифрованим текстом припускають здатність супротивника встановити зв'язок із ціллю. Це застосовується у випадку, коли зловмисник може звертатися до цільового пристрою декапсуляції за допомогою обраних ним зашифрованих текстів.

1.1. Атаки відновлення ключа – відомий шифротекст

Атаки відновлення ключа у випадку з відомим шифротекстом зазвичай спрямовані на витік із операцій, які безпосередньо маніпулюють секретним модулем s в рамках процедури декапсуляції. Поліноміальне множення на основі NTT, що використовується в процедурі дешифрування, можна використовувати для відновлення ключа. Primas та ін. у роботі [3] представили SCA, націлену на NTT, покладаючись на методи на основі Soft-Analytical Side-Channel Attack (SASCA) [4] для відновлення ключа. Їм вдалося відновити весь ключ в одному захопленому сліді бічного каналу (Power/EM) зі схеми Ring-LWE, що працювала на мікроконтролері ARM Cortex-M4. Атака була спрямована на витік з операції $INTT(\hat{u}' \circ s)$ у процедурі дешифрування. Метою є відновлення вхідних даних $(\hat{u}' \circ s)$, що призводить до відновлення приватного ключа s.

Атака проходить у дві фази. По-перше, фаза профілювання використовується для створення шаблонів для проміжних операцій. На етапі атаки ці шаблони зіставляються з відповідними сегментами в отриманому сліді атаки, а результати об'єднуються за допомогою відомого алгоритму поширення переконань (Belief Propagation) [5] для відновлення приватного ключа.

У атаки є кілька недоліків [2]:

- необхідність тривалого етапу профілювання (з понад 100 мільйонами шаблонів;
- детальне знання реалізації INTT;
- вимога відносно високого відношення «сигнал-шум» (SNR) для успішного відновлення ключа.

1.2. Атаки відновлення ключа – обраний шифротекст

Зловмисник може створювати обрані зашифровані тексти, які під час декапсуляції мають здатність посилювати секретно-залежний витік з кількох операцій у рамках процедури декапсуляції. Далі наведено різні типи CCA (атак з обраним шифротекстом), які є застосовними для відновлення ключа [2].

Атаки бічними каналами на основі Оракула.

Це основна категорія атак відновлення ключа у випадку атак з обраним шифротекстом. Вони застосовуються наступним чином: зловмисник робить запит на процедуру декапсуляції за допомогою власноруч створених шифротекстів. Ці шифротексти створені таким чином, що розшифроване повідомлення m' дуже тісно пов'язане з цільовою частиною секретного ключа або, у деяких випадках, із усім секретним ключем. Зловмисник використовує витік із операцій, що займаються обробкою розшифрованого повідомлення, щоб відновити його, таким чином реалізуючи практичний оракул бічного каналу. Така інформація, отримана через кілька ретельно створених зашифрованих текстів, розкриває повний приватний ключ.

Нижче наведено три основні підкатегорії CCA (атак з обраним шифротекстом) з використанням бічних каналів на основі оракула [2]:

1. Атаки бічними каналами з перевіркою відкритого тексту на основі Оракула (Оракул_ПВТ).

Основна ідея цих атак полягає в створенні зашифрованих текстів, щоб обмежити кількість можливих розшифрованих повідомлень. Крім того, значення розшифрованого повідомлення також залежить від єдиного цільового коефіцієнта секретного ключа для вибраних зашифрованих текстів. Витік із бічного каналу з операцій, що займаються обробкою повідомлення використовується для створення екземпляра оракула перевірки відкритого тексту (PC) для відновлення ключа. D'Anvers та ін. у роботі [6] представили атаку бічними каналами з перевіркою відкритого тексту на основі оракулу на схеми PQC, такі як LAC і RAMSTAKE, яка використовує часовий бічний канал для кодів виправлення помилок з непостійним часом. Згодом Ravi на ін. у роботі [7] узагальнили атаку на всі KEM на основі LWE/LWR у другому раунді процесу NIST, включаючи Kyber KEM, використовуючи бічний канал електромагнітної енергії.

Повне відновлення ключа можна виконати за $\approx 2k-4k$ запити для всіх наборів параметрів Kyber KEM. Однією з головних переваг є те, що атака може використовувати витік із усієї процедури повторного шифрування (рядок 19), і, отже, може працювати в налаштуваннях низького відношення «сигнал-шум» і недорогому обладнанні для атак бічними каналами [2].

2. Атаки бічними каналами шляхом збою розшифрування на основі Оракула (Оракул_ЗР)

Ця категорія атак працює шляхом виконання запитів до пристрою декапсуляції за допомогою обережно пошкоджених зашифрованих текстів, так що викликані через пошкодження збої розшифрування залежать від секретного ключа. Тому оракул бічного каналу, який здатний виявити помилки дешифрування, може відновити секретний ключ. Першу подібну атаку бічним каналом шляхом збою розшифрування на основі оракула було запропоновано у роботі Guo та ін. [8] на Frodo KEM, використовуючи непостійне за часом виконання операції порівняння зашифрованого тексту для виявлення помилок дешифрування. Згодом Bhasin та ін. [9] узагальнили атаку на Kyber KEM і продемонстрували, що витік потужності/електромагнітного випромінювання з блоку порівняння зашифрованого тексту може бути використаний для виявлення збоїв дешифрування для відновлення ключа.

Кілька робіт [10, 11] показують, що простого відновлення знака вихідного шуму $d[0]$, тобто $d[0]>0$ або $d[0]<0$ для кількох дійсних шифротекстів достатньо для того, щоб повністю відновити секретний ключ. Помилка дешифрування може бути легко ідентифікована за допомогою витоку бічними каналами з будь-якої операції в рамках процедури повторного шифрування, а також операції порівняння зашифрованого тексту.

3. Атаки бічними каналами шляхом повного розшифрування на основі Оракула (Оракул_ПР).

Щодо цього Xu та ін. у роботі [12] запропонували нову техніку побудови обраних зашифрованих текстів, які одночасно надають 256 біт інформації про приватний ключ. У роботі створюється $ct=(u,v)\in(R_q^k\times R_q)$ так, що

$$u_i=\begin{cases}U\cdot x^0 & \text{if } \ i=0\\ 0 & \text{if } \ 1\le i\le k\text{-}1\end{cases}, \tag{1}$$

$$v=V\cdot\left(\sum_{i=0}^{i=n-1}x^i\right), \tag{2}$$

де $(U,V)\in\square^+$. Зловмисник може вибрати кортежі (U,V) так, що розшифроване повідомлення матиме наступний вигляд:

$$m_i'=F(s_0[i]) \qquad \text{if } \ 1\le i\le n\text{-}1 \tag{3}$$

де кожен біт повідомлення m_i' залежить від відповідного секретного коефіцієнта $s_0[i]$. Крім того, зловмисник може вибрати (U,V) так, щоб кожен біт повідомлення m_i' однозначно ідентифікував відповідний секретний коефіцієнт $s_0[i]$. Таким чином зловмисник ефективно розпаралелив атаку Оракул_ПВТ. Оскільки для відновлення ключа потрібен доступ до повного розшифрованого повідомлення (тобто до оракула повного розшифрування), у роботі [13] пропонується використовувати витік з операції кодування повідомлення під час повторного зашифрування, який можна використовувати для відновлення 256 біт в одному сліді.

Таким чином, повне відновлення ключа можливе шляхом всього лише шести запитів для Kyber512. Подібним чином у роботах Ravi та ін. [14] і Ngo та ін. [15, 16] демонструється можливість використання витоку з операції декодування повідомлення у випадку з обраним шифротекстом для повного відновлення ключа приблизно в 6 – 20 слідах для таких схем, як Kyber і Sabre.

Показані атаки демонструють, що зловмисник може використовувати обрані шифротексти для отримання витоку з різних операцій у рамках процедури декапсуляції для відновлення ключа.

Націлювання на операцію NTT (Витік_з_NTT).

Хоча, для атак з відомим шифротекстом подібні атаки можуть витримувати лише шум зі стандартним відхиленням σ в діапазоні 0,5 – 0,7. Нещодавно Hamburg та ін. [17] про демонстрували, що чутливість цих атак до відношення «сигнал-шум» (SNR) може бути значно покращено у випадку з обраним шифротекстом.

Ідея полягала у створенні обраних шифротекстів, щоб подавати рідкісні вхідні дані $(\hat{u}' \circ \hat{s})$ до екземпляра INTT у процедурі дешифрування. Повідомляється, що це покращує ефективність алгоритму поширення переконань (Belief Propagation), дозволяючи більше шуму під час вимірювань, навіть при націленні на замасковані реалізації. Вони демонструють низку атак відновлення ключа зі складністю сліду в діапазоні від k до $2k$, де k є розміром модуля в Kyber KEM ($k = \{2,3,4\}$). Покращена атака може витримувати набагато більше шуму з $\sigma \leq 2.2$, демонструючи значне покращення атак NTT, якщо вони виконуються у випадку з обраним шифротекстом

1.3. Атаки відновлення повідомлення – відомий шифротекст

Подібні атаки можна розділити на дві категорії:

Націлювання на операції кодування та декодування повідомлень (Витік_з_Кодування_Декодування).

Атаки відновлення повідомлень переважно спрямовані на дві операції, які безпосередньо виконують дії над конфіденційним повідомленням m: операцію $Encode$ в зашифруванні та операцію $Decode$ в розшифруванні. І операції кодування, і операції декодування виконують дії над повідомленням по одному біту за раз, і це побітове маніпулювання конфіденційним повідомленням служить основним джерелом витоку для таких атак. Першу таку атаку продемонстрували Amiet та ін. [18], спрямована вона на операцію кодування повідомлень у NewHope KEM, KEM на основі Ring-LWE на мікроконтролері ARM Cortex-M4. Різницю у вазі Хеммінга коефіцієнтів полінома повідомлення $m[i] = [q/2]$ для $m_i = 1$ та $m[i] = 0$ для $m_i = 0$ можна легко розрізнити за допомогою атаки бічним каналом, що дозволяє повністю відновити окремі біти повідомлення в одному сліді. Згодом Sim та ін. [9] узагальнили техніку атаки, щоб націлити її на всі KEM на основі решітки, що беруть участь у процесі стандартизації NIST, включаючи Kyber KEM.

Пізніше у роботі [8] було представлено нові атаки, які використовують витік з операції декодування повідомлення в процедурі розшифрування для відновлення повідомлення. Незважаючи на те, що в роботі було продемонстровано наявність витоку з окремих бітів повідомлення, вдалося отримати лише 81 % успіху для відновлення одного байта повідомлення Kyber, тоді як обладнання з більш високим SNR потенційно могло б виконати ідеальне відновлення повідомлення з одного сліду.

Націлювання на операцію NTT (Витік_з_NTT).

У роботі [19] демонструється, що операція NTT також може бути ціллю для атаки відновлення повідомлень. Ідея полягала у тому, щоб відновити вхідні дані для екземпляру NTT через ефемерний секрет r, знання якого можна використати для відновлення повідомлення m із шифротексту ct. У роботі [19] також пропонується значне покращення оригінальної атаки з роботи [3] шляхом зменшення кількості шаблонів з одного мільйона до лише 213 шаблонів, а також надання кількох вдосконалень, таких як використання вдосконаленого алгоритму поширення переконань (Belief Propagation) для відновлення повідомлень. Було також показано, що ця атака може бути застосована навіть при використанні маскувальних контрзаходів, хоча й за наявності високого відношення «сигнал-шум».

1.4. Атаки відновлення повідомлення – обраний шифротекст

Націлювання на захищені операції кодування та декодування повідомлень (Витік_з_захищеного_Кодування_Декодування)

Атаки відновлення повідомлень, націлені на операції кодування та декодування повідомлень, здатні відновити все повідомлення в одному сліді у випадку атак з відомим шифротекстом. Однак подібним атакам можна легко запобігти за допомогою простих контрзаходів перетасування, які рандомізують порядок кодування/декодування окремих бітів повідомлення. Хоча перетасування не усуває джерело витоку бічним каналом, воно не дає зловмиснику можливості відновити правильний порядок бітів повідомлення, тим самим перешкоджаючи відновленню повідомлення з одного сліду.

Однак у роботі Ravi та ін. [14] показано, що зловмисник може порушити контрзахід перетасування у випадку з обраним шифротекстом, використовуючи властивість пластичності зашифрованого тексту схем на основі LWE/LWR. Враховуючи цільовий шифротекст $ct = (u, v)$, зловмисник спочатку піддає ct процедурі декапсуляції, щоб відновити окремі біти повідомлення m' через бічні канали, а потім обчислює його вагу Хеммінга (HW). Після цього зловмисник подає пошкоджений (спотворений) зашифрований текст $ct^* = (u, v + q/2 \cdot x^0)$, тобто $q/2$, доданий до першого коефіцієнта v. Це має ефект інвертування першого біта повідомлення m_0', що призводить до спотвореного повідомлення m''. Різниця в HW m' і m'' (збільшення або зменшення) може бути використана для відновлення значення інвертованого біта повідомлення m_0. Таким чином, можливе повне відновлення повідомлення за 257 запитів для Kyber KEM.

2. Атаки помилками на Kyber KEM

У даному пункті представлено короткий огляд атак із впровадженням помилок на KEM на основі структурованої решітки, з основним акцентом на атаках, які застосовуються до Kyber [2].

2.1. Атаки відновлення ключа і відновлення повідомлення – відомий шифротекст

Повторне використання nonce (Повторне_використанн_Nonce).

Ця категорія охоплює атаки на процедури генерації ключів та інкапсуляції, де зловмисник може спостерігати лише помилкові виходи від цілі. У роботі Ravi та ін. [20] запропоновано першу практичну атаку помилками, що застосована до KEM на основі решітки, таких як Kyber, NewHope і Frodo. Запропонована атака випливає зі спостереження, що початкове число, яке використовується для вибірки секрету, і помилки (похибки) для екземплярів LWE відрізняються лише на один байт, тобто s і e обираються з того самого початкового числа $seed_B$, але з різними nonce (випадковими числами) $coins_s$ і $coins_e$, які, в свою чергу, відрізняються одним байтом. Те ж саме стосується процедури зашифрування.

Таким чином, зловмисник може використовувати помилки для примусового повторного використання nonce, тобто так, що $coins_s = coins_e$, щоб створити екземпляри LWE у формі $t = A \cdot s + s = A \cdot (s + 1)$, які можна тривіально розв'язати за допомогою елімінації Гауса. Автори продемонстрували практичність повторного використання nonce за допомогою введення (ін'єкції) електромагнітного збою (EMFI) на мікроконтролері ARM Cortex-M4. У той час, як атака призводить до повного відновлення ключа та відновлення повідомлення у випадку атаки «Людина посередині» (MITM), вона вимагає введення кількох цільових помилок до процедур генерації ключів та інкапсуляції для практичних атак.

У роботі [21] Valencia та ін. провели більш загальне дослідження сприйнятливості CPA-захищених схем на основі LWE/LWR до атак помилками. Вони пропонують різні атаки, націлені на кілька операцій у рамках процедур генерації ключів, шифрування та дешифрування. Однак ці атаки розглядають моделі помилок, такі як обнулення цілих поліномів, чого важко досягти на практиці.

2.2. Атаки відновлення ключа – обраний шифротекст

Однією з головних проблем, пов'язаних із націлюванням на процедуру декапсуляції шляхом помилок, є те, що вона містить внутрішній захист від помилок, тобто FO-перетворення (Fujisaki-Okamoto transformation) для виявлення недійсних зашифрованих текстів із дуже високою ймовірністю. Це представляє значну проблему для виконання атак помилками.

Націлювання на перевірку рівності шифротексту (Пропуск_Перевірки_Шифротексту).

Однією з очевидних цілей в процесі виконання процедури декапсуляції є пропуск перевірки рівності зашифрованого тексту шляхом помилок. У роботі [22] було показано, що безпеку відносно CCA кількох схем, включаючи Kyber, можна легко зламати через одну цільову помилку. Аналіз реалізації операції перевірки рівності зашифрованого тексту в програмній реалізації Kyber KEM з бібліотеки pqm4 [23] показує наступне: масив T містить конфіденційний сумісно використовуваний секрет $\bar{K}'$, отриманий із розшифрованого повідомлення m' після розшифрування. Якщо порівняння зашифрованого тексту не вдається (недійсний/зловмисний зашифрований текст), псевдовипадкове значення z записується в T за допомогою операції умовного переміщення. Згодом T використовується для отримання остаточного спільного секрету K.

Таким чином, процедура декапсуляції записує конфіденційний сумісно використовуваний секрет до T (за умови успішної декапсуляції), перш ніж перевірити дійсність зашифрованого тексту. Таким чином, простий пропуск наступної операції умовного переміщення гарантує використання конфіденційного сумісно використовуваного секрету $\bar{K}'$ для генерування спільного секрету K, навіть якщо операція порівняння зашифрованого тексту дала збій. У роботі [22] показано, що така вразливість може бути використана через прості збої у частоті і згодом може призвести до відновлення ключа за рахунок кількох тисяч запитів із застосуванням атаки обраного зашифрованого тексту.

Атака помилками з обраним шифротекстом.

За винятком перевірки рівності зашифрованого тексту, у процедурі декапсуляції немає інших тривіальних цілей для помилок. Проте Pessl та Prokop у роботі [24] запропонували першу загальну атаку помилками з обраним шифротекстом, яка працює шляхом введення цільових помилок в операцію декодування повідомлення в рамках процедури дешифрування, так що результуючий успіх/невдача декапсуляції може бути використаний для отримання важливої інформації про приватний ключ.

Основна ідея атаки полягає в наступному. Зловмисник надсилає дійсний зашифрований текст ct для декапсуляції та вводить одну помилку, щоб пропустити додавання з $q/2$ під час декодування одного коефіцієнта полінома повідомлення $m'[i]$. Це має непрямий ефект спотворення $m'[i]$ приблизно на $q/4$. Це призводить до інверсії m'_i (збій декапсуляції) лише тоді, коли відповідний коефіцієнт шумового компонента $d[i] < 0$. Однак, коли $d[i] \geq 0$ у m'_i немає жодних змін. Це допомагає зловмиснику створити єдину лінійну нерівність, використовуючи $d[i]$, а зловмисник, який здатний побудувати $5k - 7k$ таких лінійних нерівностей, може виконати повне відновлення ключа для Kyber KEM.

Хоча атаку було продемонстровано за допомогою збою частоти на мікроконтролері ARM Cortex-M4, для атаки все ще потрібно вводити цільову помилку пропуску в процедуру декодування повідомлення. Таким чином, цій атаці можна запобігти, просто застосувавши перемішування для операції декодування повідомлення.

Згодом Hermelink та ін. у роботі [11] запропонували покращення атаки з роботи [24], застосувавши дещо інший підхід. Замість того, щоб використовувати дійсний зашифрований текст ct, автори пропонують подати спотворені зашифровані тексти так, що один коефіцієнт другого компонента зашифрованого тексту $v[i]$ спотворюється на $q/4$. Під час подання спотвореного зашифрованого тексту після дешифрування вводиться помилка, щоб виправити однобітове спотворення в зашифрованому тексті, що зберігається в пам'яті. Якщо внесене спотворення призвело до правильного дешифрування ($d[i] \geq 0$), то введена помилка виправ-

ляє спотворення в зашифрованому тексті, забезпечуючи успішну декапсуляцію. Однак, якщо початкове спотворення призвело до помилки дешифрування ($d[i] < 0$), то це призводить до помилки декапсуляції, навіть після виправлення спотворення в збереженому шифротексті шляхом введення помилки. Ця інформація про d, отримана приблизно за $5k-7k$ таких запитів, може відновити повний приватний ключ.

На відміну від атаки з [24], атака з [11] не має часових обмежень для ін'єкції (введення) помилки, оскільки їй потрібно лише ввести помилку інверсії біта в пам'яті в будь-який час між операцією дешифрування та порівняння зашифрованого тексту. Проте введення точних одиночних помилок інверсії бітів у пам'яті потребує детальної інформації про цільовий пристрій, а також про реалізацію та розширене профілювання цільового пристрою. Нещодавно Del-vaux [25] покращив атаку [11] шляхом розширення поверхні атаки до кількох операцій у рамках процедури декапсуляції, а також роботи з різноманітними більш вільними моделями помилок, такими як довільна інверсія бітів, помилки встановлення у 0, випадкові помилки та помилки пропуску інструкцій. Однак атаки, що покладаються на більш вільну модель помилки, можуть вимагати понад $100k$ запитів з обраним зашифрованим текстом для повного відновлення ключа, залежно від практичності моделі помилки.

На рис. 1, 2 наведено всі SCA та FIA відповідно, що застосовуються для Kyber KEM [2]:

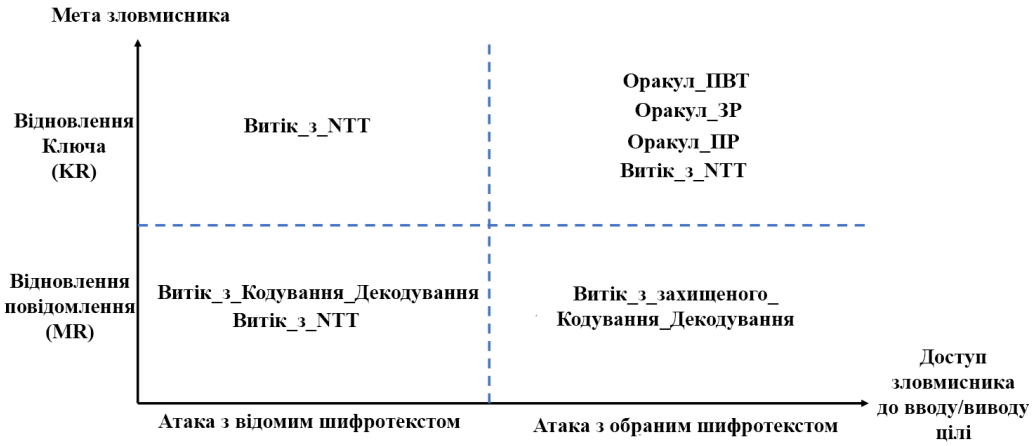

Рис. 1. Атаки бічними каналами (SCA) на Kyber KEM

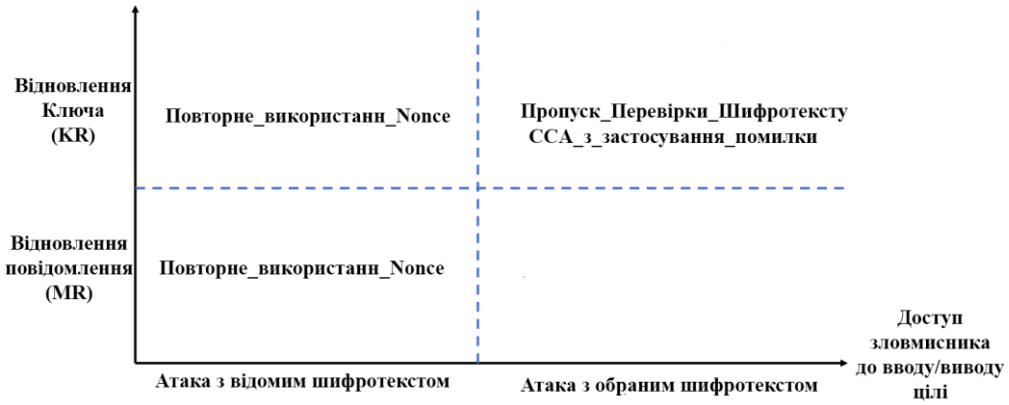

Рис. 2. Атаки помилками (FIA) на Kyber KEM

3. Захист Kyber KEM від атак бічними каналами та помилками

У попередньому розділі було представлено детальний огляд різних SCA та FIA на Kyber KEM. У цьому розділі будуть надані ряд контрзаходів, які можна використовувати для захисту від згаданих атак [2].

3.1. Захист від атак бічними каналами та помилками з обраним шифротекстом

Підхід ґрунтується на стратегії виявлення, щоб перевірити/визначити, чи є отриманий зашифрований текст шкідливим. У разі виявлення шкідливого/зловмисного тексту ціль може просто відхилити зашифрований текст і змінити/оновити пару відкритий-приватний ключ, повторно запустивши процедуру генерації ключа. Перевагою цього підходу є те, що таке виявлення запобігає подальшому розкриттю приватного ключа.

Перевірка нормальності шифротексту.

Детальний аналіз шифротекстів, які використовуються в Оракул_ПВТ і Оракул_ПР, показує, що більшість коефіцієнтів зашифрованого тексту мають фіксоване значення 0. Однак коефіцієнти дійсного зашифрованого тексту рівномірно розподілені в діапазоні $[0, q]$, враховуючи, що обидва компоненти зашифрованого тексту є по суті екземплярами LWE. Тому пропозицією є проведення статистичного аналізу зашифрованого тексту перед використанням у процедурі дешифрування.

У якості статистичних даних для виявлення перекосу в отриманому зашифрованому тексті було вирішено обрати середнє значення та стандартне відхилення коефіцієнтів зашифрованого тексту. Для заданого полінома $x \in R_q$ позначаємо середнє (μ) і стандартне відхилення (σ) коефіцієнтів x як $\mu(x)$ і $\sigma(x)$ відповідно. Було виконане емпіричне моделювання, щоб обчислити середнє значення та стандартне відхилення $\mu(u)$ і $\sigma(u)$ для окремих поліномів компонента зашифрованого тексту u, а також $\mu(v)$ і $\sigma(v)$ для компонента зашифрованого тексту v, що відповідає дійсним шифротекстам для Kyber KEM. Отримані значення для середнього та стандартного відхилення всіх чотирьох статистичних показників:

$$(\mu(\mu(u)), \sigma(\mu(u))) = (1663, 60) \quad (4)$$

$$(\mu(\mu(u)), \sigma(\mu(u))) = (959, 27)$$

$$(\mu(\mu(v)), \sigma(\mu(v))) = (1560, 60) \quad (5)$$

$$(\mu(\mu(v)), \sigma(\mu(v))) = (957, 27)$$

На основі стандартного відхилення σ для кожного з цих показників можна вибрати прийнятний діапазон для них. Наприклад, якщо вибрано довжину $6 \cdot \sigma$, то прийнятний діапазон для $\mu(u)$ становить $[\mu(\mu(u)) + 6 \cdot \sigma, \mu(\mu(u)) - 6 \cdot \sigma]$. Чим менший допустимий діапазон, тим вища ймовірність хибних спрацьовувань, тобто визначення дійсного зашифрованого тексту недійсним. Однак великий допустимий діапазон збільшує ймовірність пропусків, що призводить до прийняття спотвореного зловмисного зашифрованого тексту як дійсного.

За допомогою емпіричного моделювання було визначено, що довжина 6σ як для середнього, так і для стандартного відхилення призводить до ймовірності $\approx 2^{-22}$ для відхилення правильного зашифрованого тексту.

Хоча такий контрзахід здатний виявляти перекошені зашифровані тексти, обрані зашифровані тексти, що використовуються в Оракул_ЗР [6], а також ті, що використовуються в атаках помилками з обраним шифротекстом [11, 24, 25], є рівномірно випадковими. Це пояснюється тим, що подібні атаки передбачають додавання невеликих помилок до одного коефіцієнта дійсного зашифрованого тексту. Така дія не вносить помітного перекосу в коефіцієнти, тим самим перешкоджаючи контрзаходу Перевірки_нормальності_шифротектсу.

Перевірка нормальності поліному повідомлення.

Цей контрзахід ґрунтується на аналізі коефіцієнтів полінома зашумленого повідомлення $m' = (v' - u' \cdot s)$, отриманого під час дешифрування отриманого зашифрованого тексту ct. Для дійсних зашифрованих текстів ми спостерігаємо, що коефіцієнти m' розподіляються відповідно до дуже вузького розподілу Гауса поблизу $q/2$ або 0, тобто $m[i] = q/2 \pm \delta$ для $m_i = 1$ і $m[i] = 0 \pm \delta$ для $m_i = 1$.

У той час як атака помилками Pessl та ін. [26] додає $q/4$ до $m'[i]$ через помилки, атаки [9, 11, 25] явно додають $q/4$ до цільового коефіцієнта дійсного зашифрованого тексту. Таким чином, усі ці атаки працюють шляхом прямого/опосередкованого додавання $q/4$ до одного з цільових коефіцієнтів полінома повідомлення $m'[i]$. Це гарантує, що принаймні один поліном повідомлення, тобто $m'[i]$ не буде в межах очікуваного діапазону, що відповідає діапазону дійсного зашифрованого тексту

На основі цього спостереження можна запропонувати перевірити розподіл коефіцієнтів полінома повідомлення для отриманого зашифрованого тексту. Нехай допустимий діапазон буде $(q/2 \pm L \cdot \sigma)$ і $(0 \pm L \cdot \sigma)$, де $L \in \mathbb{Z}^+$ залишається на вибір розробника. Чим більший діапазон прийняття $L \cdot \sigma$, тим меншою є ймовірність відкинути дійсний зашифрований текст. Однак вибір меншого діапазону підвищує шанси невиявлення зловмисного обраного зашифрованого тексту. Тому для покращення безпеки важливо вибрати правильне значення для L. При $L = 6$ після більш ніж 2^{25} дійсних декапсуляцій не було виявлено хибнопозитивного результату [2].

Якщо є принаймні один коефіцієнт за межами цього допустимого діапазону, ми просто позначаємо зашифрований текст як дійсний і оновлюємо пару відкритий-приватний ключ. Хоча цей контрзахід вимагає розшифрувати принаймні один обраний шифртекст для успішного виявлення, однак CCA вимагають принаймні від кількох десятків до кількох тисяч запитів для відновлення ключа.

3.2. Захист операції порівняння шифротексту (Захист_Порівняння_Шифротексту)

У даному пункті пропонуються два рівні захисту для операції порівняння зашифрованого тексту, на яку спрямована атака Пропуск_Перевірки_Шифротексту Xagawa та ін. [22]. На першому рівні додається захист від пропуску операції порівняння зашифрованого тексту, для чого використовується лічильник динамічного циклу, щоб відстежувати кількість порівнюваних байтів зашифрованого тексту в операції порівняння зашифрованого тексту. Якщо кількість байтів зашифрованого тексту для порівняння дорівнює d, тоді лічильник циклу l ініціалізується випадковим значенням $k \cdot d$, де $k \in \mathbb{Z}^+$ вибирається випадковим чином під час кожного виконання. Потім l зменшується на k для кожного байта зашифрованого тексту, що порівнюється. Таким чином, якщо $l = 0$ після порівняння зашифрованого тексту, то можна впевнитись, що всі байти були порівняні. Використання такого динамічного лічильника циклу додає додатковий рівень захисту від атак помилками вищого порядку, які також намагаються збити лічильник циклу.

Щоб захиститись від пропуску операції умовного переміщення, слід записувати спільно використовуваний секрет $\bar{K}'$ в тимчасову змінну tmp замість T. $\bar{K}'$ копіюється в T, якщо порівняння зашифрованого тексту вдалось ($l = 0$), у іншому випадку z копіюється в T. Перевірка того, чи $l = 0$ виконується для кожного байта, скопійованого в T. Це просте виправлення реалізації гарантує, що пропуск операції умовного переміщення не розкриє жодної інформації про спільно використовуваний секрет $\bar{K}'$ для недійсних зашифрованих текстів, тим самим перешкоджаючи відновленню ключа.

3.3. Захист операцій кодування/декодування повідомлення (Перемішування_Кодування_Декодування)

SCA, що націлені на процедури кодування (Encode) і декодування (Decode) повідомлення, є дуже потужними, враховуючи, що зловмисник може виконати відновлення повідомлення в одному сліді. Крім того, витік від цих операцій також можна використовувати як Оракул_ПР для відновлення ключа [12]. Ravi та ін. у роботі [14] показали, що контрзаходи

перетасування для операцій кодування та декодування повідомлень можуть бути зламані у випадку статичного ключа через CCA за кількість від кількох сотень до кількох тисяч запитів. Хоча перетасування не забезпечує конкретного захисту, воно обґрунтовано збільшує зусилля зловмисника щодо відновлення повідомлення, як показано в [16]. Крім того, у ефемерних умовах перетасування забезпечує конкретний захист, оскільки всі вищезгадані атаки вимагають від кількох сотень до кількох тисяч запитів для відновлення повідомлення [14]. Таким чином, реалізація контрзаходу перетасування для операцій кодування та декодування Kyber KEM не буде зайвою.

3.4. Захист NTT (Перемішування_NTT, Маскування_NTT)

SCA, націлені на NTT, також здатні виконувати відновлення ключа та повідомлення в одному сліді [3, 19] або дуже невеликій кількості слідів [17] (атаки Витік_з_NTT). У роботі [26] запропоновано низку загальних контрзаходів перетасування та маскування з різним ступенем деталізації для захисту NTT від вищезгаданих атак із одним слідом. Ґрунтуючись на передбачуваному рівні загрози від потенційного зловмисника та прийнятній продуктивності, розробник може вибрати відповідний контрзахід перетасування для операції NTT.

3.5. Захист вибірки секретів і помилок (похибок) (Збиткове_Порівняння)

У роботі [20] продемонстровано, що повторне використання nonce може бути викликане помилками в процедурі генерації ключа та шифрування Kyber KEM для атак відновлення ключа та відновлення повідомлення відповідно. Таким чином, тривіальним захистом від цієї атаки може бути виконання надлишкового обчислення процедури вибірки. Хоча це не забезпечує повного захисту, це значно підвищує складність для зловмисника.

4. Експериментальна оцінка

У цьому пункті показано практичну оцінку ефективності представлених контрзаходів при інтеграції в оптимізовану реалізацію програмного забезпечення Kyber, що працює на мікроконтролері STM32F4 на основі процесора ARM Cortex-M4.

4.1. Цільова платформа та деталі реалізації

Цільовою платформою для процесора ARM Cortex-M4 є плата STM32F4DISCOVERY, на якій розміщено мікроконтролер STM32F407, а тактова частота становить 24 МГц. Контрзаходи були реалізовані на оптимізованій для M4 реалізації Kyber, доступній у загальнодоступній бібліотеці pqm4 [23] – середовищі порівняльного аналізу схем PQC на мікроконтролері ARM Cortex-M4. Оптимізована для M4 реалізація Kyber базується на високошвидкісній реалізації з ефективним використанням пам’яті, запропонованою Ботросом, Каннвішером і Швабе в [27]. Усі реалізації було скомпільовано за допомогою компілятора arm-none-eabi-gcc-7.3.1 з використанням прапорів компілятора -O3 -mthumb -mcpu=cortex-m4 -mfloat-abi=hard -mfpu=fpv4-sp-d16

4.2. Результати експериментів

У табл. 1 представлено дані про рівень витрат продуктивності через заходи протидії перемішування та маскування (Перемішування_NTT, Маскування_NTT) проти атак витоку з NTT (Витік_з_NTT) на Kyber KEM, який працює на пристрої ARM Cortex-M4. Подібні експерименти надані у роботі [2], головною метою було відтворення та перевірка отриманих у роботі результатів. Н/з – незахищена реалізація, З – реалізація з контрзаходами, %В – рівень затрат продуктивності у відсотках.

Таблиця 1

Продуктивність контрзаходів Shuffled_Masked_NTT для Kyber KEM порівняно з оптимізованими незахищеними реалізаціями на пристрої ARM Cortex-M4. Дані отримано на мікроконтролері STM32F407VG, встановленому на платі STM32F407DISCOVERY, що працює на частоті 24 МГц. Числа позначають кількість тактових циклів $\times 10^3$

Схема	Такти ($\times 10^3$)								
	KeyGen			Encaps			Decaps		
	Н/з	З	%В	Н/з	З	%В	Н/з	З	%В
Контрзахід перемішування									
Kyber512	457.3	782.4	71.1	552.1	970.6	75.8	511.3	1022.6	100.0
Kyber768	748.8	1237.7	65.3	903.6	1483.7	64.2	842.5	1506.4	78.8
Kyber1024	1188.2	1841.5	54.9	1378.5	2125.6	54.2	1297.5	2125.9	63.9
Контрзахід маскування									
Kyber512	457.3	729.4	59.5	552.1	898.2	62.7	511.3	933.6	82.6
Kyber768	748.8	1155.4	54.3	903.6	1386.1	53.4	842.5	1399.4	66.1
Kyber1024	1188.2	1732.4	45.8	1378.5	1998.8	45.0	1297.5	1993.0	53.6

На пристрої ARM Cortex-M4 можна спостерігати вплив на продуктивність у діапазоні 46 – 71 % для генерації ключів, 45 – 76 % для інкапсуляції та 53 – 100 % для процедури декапсуляції для всіх наборів параметрів Kyber KEM.

У табл. 2 наведено дані про рівень витрат продуктивності через контрзаходи Перевірка_нормальності_шифротексту, Перевірка_нормальності_поліному_повідомлення, Захист_Порівняння_Шифротексту і Перемішування_Кодування_Декодування для Kyber KEM, який працює на пристрої ARM Cortex-M4 [2]. Н/з – незахищена реалізація, З – реалізація з контрзаходами, %В – рівень затрат продуктивності у відсотках.

Таблиця 2

Продуктивність спеціальних контрзаходів SCA-FIA для Kyber KEM порівняно з оптимізованою незахищеною реалізацією на пристрої ARM Cortex-M4. Дані отримано на мікроконтролері STM32F407VG, встановленому на платі STM32F407DISCOVERY, що працює на частоті 24 МГц. Числа позначають кількість тактових циклів $\times 10^3$

Схема	Такти ($\times 10^3$)		
	Decaps		
	Н/з	З	%В
Перевірка нормальності шифротексту			
Kyber512	511.4	689.4	34.8
Kyber768	842.4	1029.4	22.2
Kyber1024	1298.1	1492.8	15.0
Перевірка нормальності поліному повідомлення			
Kyber512	511.4	676.0	32.2
Kyber768	842.4	1005.8	19.4
Kyber1024	1298.1	1474.7	13.6
Захист операції порівняння шифротексту			
Kyber512	511.4	579.4	13.3
Kyber768	842.4	909.8	8.0
Kyber1024	1298.1	1364.3	5.2
Захист операцій кодування/декодування повідомлення			
Kyber512	511.4	520.1	1.7
Kyber768	842.4	854.2	1.4
Kyber1024	1298.1	1312.4	1.1

Як видно з табл. 2, на пристрої ARM Cortex-M4 ці контрзаходи створюють дуже розумний рівень витрат продуктивності в діапазоні 15 – 35 %, 13 – 32 %, 5 – 13 % і 1 – 2 % для різних наборів параметрів Kyber KEM.

Отримані дані показано у вигляді діаграми на рис. 3 – 8.

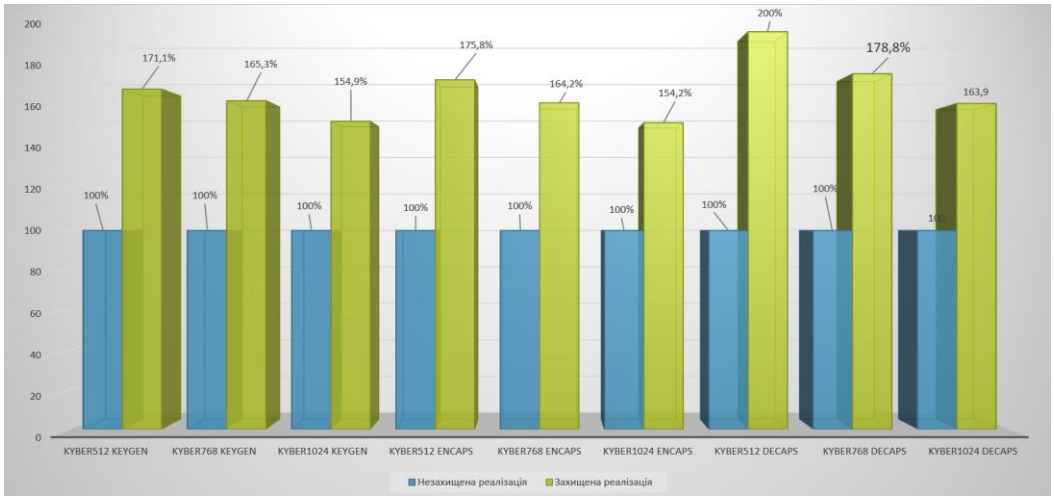

Рис. 3. Оцінка контрзаходу перемішування

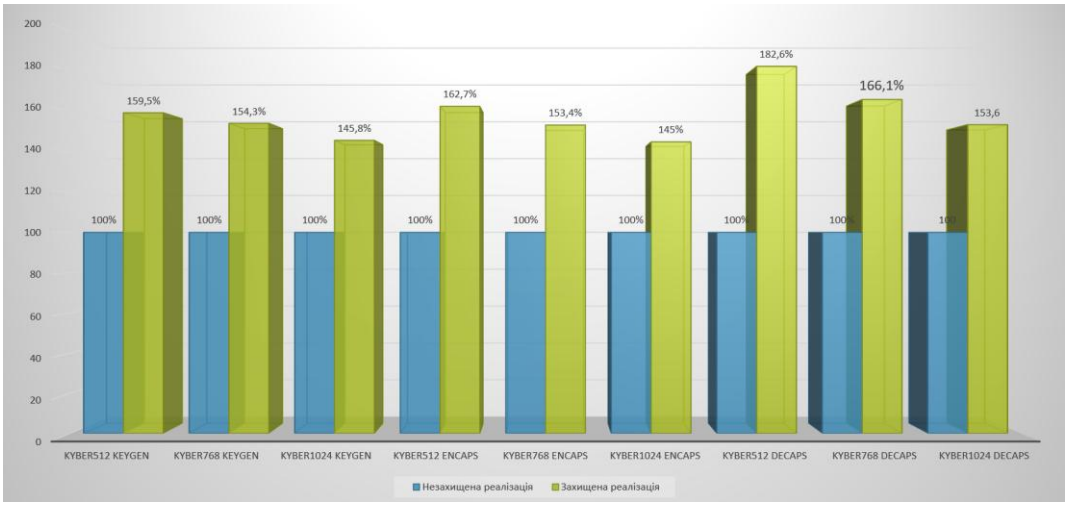

Рис. 4. Оцінка контрзаходу маскування

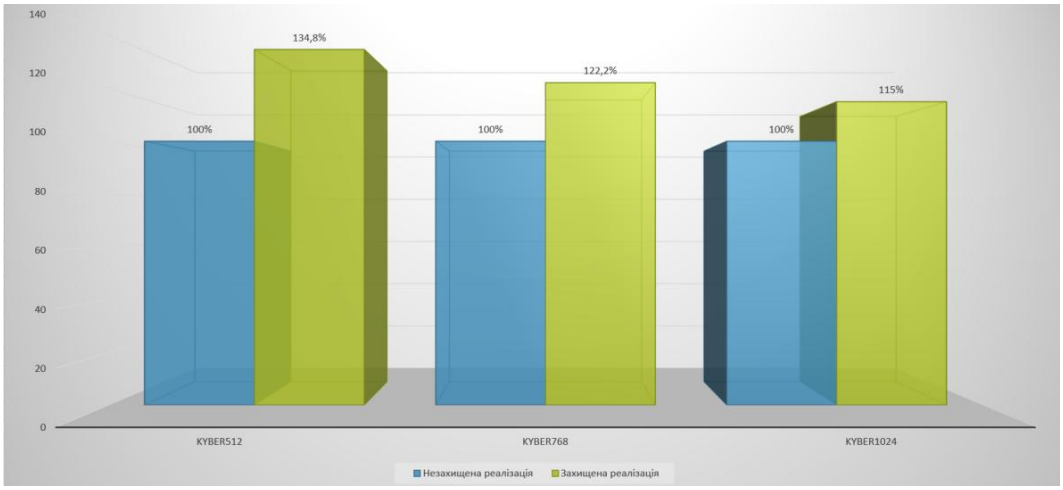

Рис. 5. Оцінка контрзаходу перевірки нормальності шифротексту

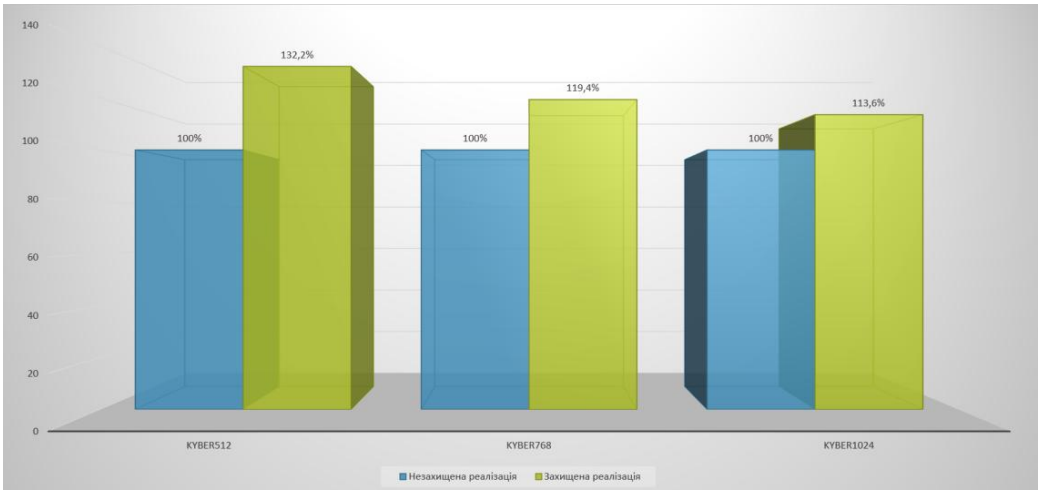

Рис. 6. Оцінка контрзаходу перевірки нормальності поліному повідомлення

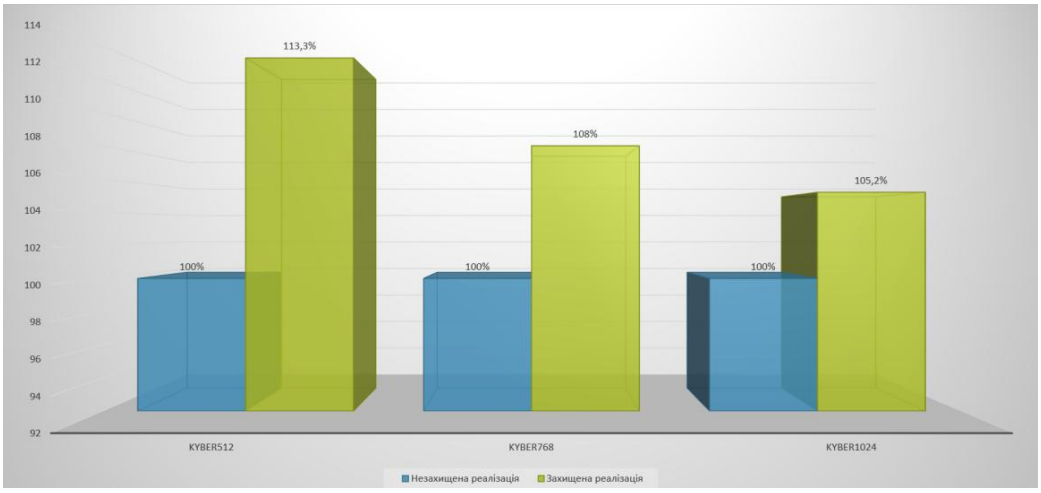

Рис. 7. Оцінка контрзаходу захисту операції порівняння шифротексту

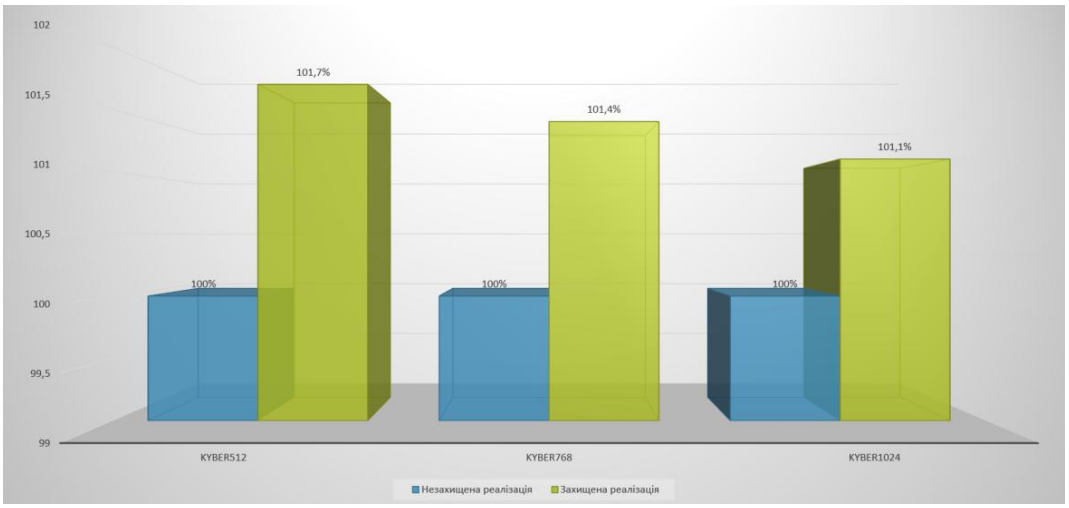

Рис. 8. Оцінка контрзаходу захисту операцій кодування/декодування повідомлення

Зауважимо, що оптимізовані реалізації NTT на цільових пристроях реалізовано на чистому ассемблері (оптимізовано для M4 та оптимізовано для NEON), але контрзаходи реалізовано на основі реалізацій NTT/INTT на основі C, що призводить до високих витрат продук-

тивності. Таким чином, можна отримати значно менші витрати продуктивності за умови, що захищені NTT/INTT реалізовані на ассемблері.

5. Порівняння рівнів захищеності CRYSTALS-KYBER та СКЕЛЯ (ДСТУ 8961-2019)

У табл. 3 надана оцінка рівнів захищеності алгоритмів CRYSTALS-KYBER [1] та СКЕЛЯ (ДСТУ 8961-2019) [28] за шкалою NIST та за Національною шкалою.

Як видно з даних у таблиці СКЕЛЯ (ДСТУ 8961-2019) [28] надає значно вищі рівні безпеки для кожного з режимів роботи, порівняно з алгоритмом Crystal-Kyber. Також однією зі значних переваг алгоритму Скеля можна вважати забезпечення захисту від спеціальних атак.

Таблиця 3

Порівняння рівнів захищеності CRYSTALS-KYBER та СКЕЛЯ

Алгоритм	Рівень захищеності NIST	Національний рівень доказової стійкості (класична / квантова)	Забезпечення захисту від спеціальних атак
KYBER512	1	-	-
KYBER768	3	0-й рівень – 128біт / 64біт	-
KYBER1024	5	1-й рівень – 256біт / 128 біт	-
СКЕЛЯ–КЕМ 256/128	5	1-й рівень – 256біт / 128 біт	+
СКЕЛЯ–КЕМ 384/192	7	2-й рівень – 384біт / 192біт	+
СКЕЛЯ–КЕМ 512/256	9	3-й рівень – 512біт / 256біт	+

На рис. 9 показано дані з табл. 3 у вигляді діаграми.

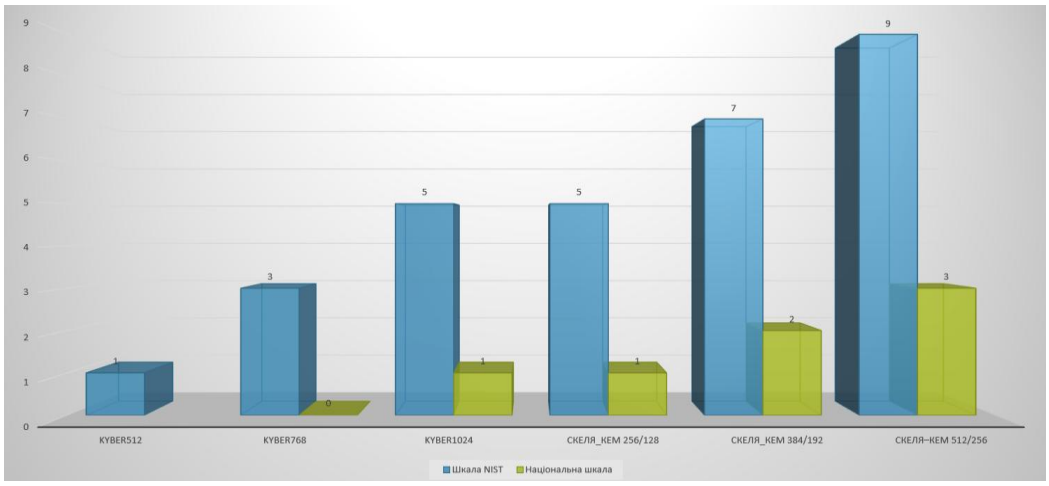

Рис. 9. Рівні безпеки алгоритмів за шкалою NIST та Національною шкалою

Висновки

Представлено систематичне дослідження атак бічними каналами (SCA) і атак помилками (із впровадженням помилок) (FIA) на структуровані схеми на основі решітки, з основним акцентом на Kyber.

Враховуючи невідповідність загальних контрзаходів, таких як маскування, для захисту від широкого спектру відомих атак, у даному пункті також представлено низку спеціальних контрзаходів для захисту від відомих SCA та FIA на Kyber.

Оцінка продуктивності показує, що представлені спеціальні контрзаходи мають розумний рівень витрат продуктивності для оцінюваної платформи. Тому, можна сказати, що у роботі наведено аргументи на користь використання спеціальних контрзаходів у реальних

реалізаціях схем на основі решітки, або окремо, або як підсилення загальних контрзаходів, таких як маскування.

Також у роботі надано оцінку рівнів захищеності алгоритмів CRYSTALS-KYBER [1] та СКЕЛЯ (ДСТУ 8961-2019) [28] за шкалою NIST та за Національною шкалою.

Як видно з даних, отриманих під час порівняння, СКЕЛЯ (ДСТУ 8961-2019) [28] надає значно вищі рівні безпеки для кожного з режимів роботи, порівняно з алгоритмом Crystal-Kyber. Також однією зі значних переваг алгоритму Скеля можна вважати забезпечення захисту від спеціальних атак.

Список літератури:

1. Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, Damien Stehlé. CRYSTALS-Kyber Algorithm Specifications And Supporting Documentation (version 3.02). URL: https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf.

2. PRASANNA RAVI, ANUPAM CHATTOPADHYAY, ANUBHAB BAKSI. Side-channel and Fault-injection attacks over Lattice-based Post-quantum Schemes (Kyber, Dilithium): Survey and New Results. 2022. URL: https://eprint.iacr.org/2022/737.pdf.

3. Robert Primas, Peter Pessl, and Stefan Mangard. Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption. 2017. URL: https://eprint.iacr.org/2017/594.pdf.

4. Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft Analytical Side-Channel Attacks. 2014. URL: https://eprint.iacr.org/2014/410.pdf.

5. Judea Pearl. Fusion, propagation, and structuring in belief networks. 1986. URL: https://ftp.cs.ucla.edu/pub/stat_ser/r42-reprint.pdf.

6. Jan-Pieter D'Anvers, Marcel Tiepelt, Frederik Vercauteren, and Ingrid Verbauwhede. Timing attacks on Error Correcting Codes in Post-Quantum Schemes. 2019. URL: https://eprint.iacr.org/2019/292.pdf.

7. Prasanna Ravi, Sujoy Sinha Roy, Anupam Chattopadhyay, and Shivam Bhasin. Generic Side-channel attacks on CCA-secure lattice-based PKE and KEM schemes. 2019. URL: https://eprint.iacr.org/2019/948.pdf.

8. Qian Guo, Thomas Johansson, and Alexander Nilsson. A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM. 2020. URL: https://eprint.iacr.org/2020/743.pdf.

9. Shivam Bhasin, Jan-Pieter D'Anvers, Daniel Heinz, Thomas Pöppelmann, and Michiel Van Beirendonck. Attacking and Defending Masked Polynomial Comparison for Lattice-Based Cryptography. 2021. URL: https://eprint.iacr.org/2021/104.pdf.

10. Jan-Pieter D'Anvers, Daniel Heinz, Peter Pessl, Michiel van Beirendonck, and Ingrid Verbauwhede. Higher-Order Masked Ciphertext Comparison for Lattice-Based Cryptography. 2021. URL: https://eprint.iacr.org/2021/1422.pdf.

11. Julius Hermelink, Peter Pessl, and Thomas Pöppelmann. Fault-enabled chosen-ciphertext attacks on Kyber. 2021. URL: https://eprint.iacr.org/2021/1222.pdf.

12. Zhuang Xu, Owen Pemberton, Sujoy Sinha Roy, David Oswald, Wang Yao, and Zhiming Zheng. Magnifying Side-Channel Leakage of Lattice-Based Cryptosystems with Chosen Ciphertexts: The Case Study of Kyber. 2020. URL: https://eprint.iacr.org/2020/912.pdf.

13. Zhuang Xu et al. Magnifying Side-Channel Leakage of Lattice-Based Cryptosystems With Chosen Ciphertexts. 2021. URL: https://ieeexplore.ieee.org/document/9591340.

14. Prasanna Ravi, Shivam Bhasin, Sujoy Sinha Roy, and Anupam Chattopadhyay. On Exploiting Message Leakage in (few) NIST PQC Candidates for Practical Message Recovery and Key Recovery Attacks. 2020. URL: https://eprint.iacr.org/2020/1559.pdf.

15. Kalle Ngo, Elena Dubrova, Qian Guo, Thomas Johansson. A Side-Channel Attack on a Masked IND-CCA Secure Saber KEM Implementation. 2021. URL: https://tches.iacr.org/index.php/TCHES/article/view/9079/8666.

16. Kalle Ngo, Elena Dubrova, and Thomas Johansson. Breaking Masked and Shuffled CCA Secure Saber KEM by Power Analysis. 2021. URL: https://eprint.iacr.org/2021/902.pdf.

17. Mike Hamburg, Julius Hermelink, Robert Primas, Simona Samardjiska, Thomas Schamberger, Silvan Streit, Emanuele Strieder, and Christine van Vredendaal. Chosen Ciphertext k-Trace Attacks on Masked CCA2 Secure Kyber. 2021. URL: https://eprint.iacr.org/2021/956.pdf.

18. Dorian Amiet, Andreas Curiger, Lukas Leuenberger, and Paul Zbinden. Defeating NewHope with a Single Trace. 2020. URL: https://eprint.iacr.org/2020/368.pdf.

19. Peter Pessl and Robert Primas. More Practical Single-Trace Attacks on the Number Theoretic Transform. 2019. URL: https://eprint.iacr.org/2019/795.pdf.

20. Prasanna Ravi, Debapriya Basu Roy, Shivam Bhasin, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Number "Not Used" Once - Practical fault attack on pqm4 implementations of NIST candidates. 2018. URL: https://eprint.iacr.org/2018/211.pdf.

21. Felipe Valencia, Tobias Oder, Tim Güneysu, and Francesco Regazzoni. Exploring the Vulnerability of R-LWE Encryption to Fault Attacks. 2018. URL: https://dl.acm.org/doi/10.1145/3178291.3178294.

22. Keita Xagawa, Akira Ito, Rei Ueno, Junko Takahashi, and Naofumi Homma. Fault-Injection Attacks against NIST's Post-Quantum Cryptography Round 3 KEM Candidates. 2021. URL: https://eprint.iacr.org/2021/840.pdf.

23. Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. PQM4: Post-quantum crypto library for the ARM Cortex-M4. 2019. URL: https://github.com/mupq/pqm4.

24. Peter Pessl and Lukas Prokop. Fault Attacks on CCA-secure Lattice KEMs. 2021. URL: https://eprint.iacr.org/2021/064.pdf.

25. Jeroen Delvaux. Roulette: Breaking Kyber with Diverse Fault Injection Setups. 2021. URL: https://eprint.iacr.org/2021/1622.pdf.

26. Prasanna Ravi, Romain Poussier, Shivam Bhasin, and Anupam Chattopadhyay. On Configurable SCA Countermeasures Against Single Trace Attacks for the NTT - A Performance Evaluation Study over Kyber and Dilithium on the ARM Cortex-M4. 2020. URL: https://eprint.iacr.org/2020/1038.pdf.

27. Leon Botros, Matthias J. Kannwischer, and Peter Schwabe. Memory-Efficient High-Speed Implementation of Kyber on Cortex-M4. 2019. URL: https://eprint.iacr.org/2019/489.pdf.

28. ДСТУ 8961:2019 Інформаційні технології. Криптографічний захист інформації. Алгоритми асиметричного шифрування та інкапсуляції ключів. URL: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=88056.

Надійшла до редколегії 07.03.2023

Відомості про авторів:

Дерев'янко Ярослав Андрійович - АТ «Інститут інформаційних технологій», науковий співробітник-консультант; Україна; e-mail: yarik0009258@gmail.com; ORCID: https://orcid.org/0000-0002-3290-3373

Горбенко Іван Дмитрович – д-р техн. наук, професор, Харківський національний університет імені В. Н. Каразіна, професор кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук, АТ "Інститут інформаційних технологій", головний конструктор; Україна; e-mail: gorbenkoi@iit.kharkov.ua; ORCID: https://orcid.org/0000-0003-4616-3449