УДК 004.056.5 **DOI:10.30837/rt.2022.3.210.03**

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

ВРАЗЛИВІСТЬ ЕП FALCON ДО СПЕЦІАЛЬНИХ АТАК ТА ЙОГО ЗАХИЩЕНІСТЬ

Вступ

Прогрес обчислювальних технологій суттєво впливає на злам існуючих стандартизованих асиметричних криптографічних перетворень. Подібно до того, як Bombe, перший електромеханічний комп'ютер зламав сумнозвісний шифр Enigma, перший практичний квантовий комп'ютер може зламати сучасні схеми асиметричного шифрування. Дійсно, добре відомо, що квантові алгоритми пропонують експоненціальне прискорення при розв'язанні завдань цілочисельної факторизації [1] та (еліптичної кривої) дискретного логарифму [2], на які покладаються існуючі системи з відкритим ключем. Для їх захисту розробляються та стандартизуються альтернативні рішення – постквантові стандарти асиметричного шифрування, протоколи інкапсуляції ключів та електронні підписи. Вони зі значною ймовірністю можуть протистояти квантовому криптоаналізу. Зростаюче занепокоєння квантовою загрозою спонукало Національний інститут стандартів і технологій (NIST) запросити та оцінити заявки на стандарт постквантової криптографії, який є постійним процесом, який планується завершити до 2023 року.

Одним із кандидатів на міжнародний стандарт електронного підпису є Falcon – алгоритм електронного підпису, що заснований на математиці алгебраїчних решіток. Недоліком проєкту стандарту електронного підпису є мала кількість досліджень стійкості проти спеціальних атак, а також атак побічними каналами [3 – 7].

У статті розглядаються та аналізуються існуючі атаки на реалізацію Falcon, а також оцінюється швидкодія при застосуванні контрзаходів, які б перешкоджали таким атакам. Незважаючи на те, що схема Falcon, а також певні математичні перетворення, все ж є вразливими до атак [8] (що в свою чергу дозволяє отримати приватний ключ), ефективність компонентів та математики даного алгоритму електронного підпису сприяє тому, що він здатен конкурувати з іншими схемами, навіть з контрзаходами проти спеціальних атак. Виявлено, що для схеми Falcon контрзаходи мають, в середньому, лише до 5 % впливу на ефективність.

1. Часові атаки на схему Falcon

Певні компоненти схеми ЕП Falcon мають вразливість до атак, що базуються на аналізі часових показників. Це пояснюється складнощами в реалізації даних компонентів постійними у часі. В цьому пункті коротко описуються вразливі місця та пропонуються контрзаходи, які здатні перешкоджати часовим атакам на них.

1.1. Вразливості алгоритму Falcon

Відбірник Гауса [9, 10]

Відбірники Гауса – одне зі слабких місць, яке є схильним до часових атак на схеми, які використовують в якості основи алгебраїчні решітки. Відбірник Гауса у схемі Falcon застосовується для відбору коротких векторів у решітках.

Теоретичне перетворення числа (NTT) [11]

Через використання значної кількості модульної арифметики NTT є потенційно вразливим до часових атак, оскільки така арифметика є складною для реалізації за постійний час. Застосування NTT для схеми Falcon необхідне з метою пришвидшення множення кільцевих многочленів [12].

HashtoPoint

Функція гешування потенційно може бути вразливою до атак на аналізі часових показників, оскільки процес гешування може бути складним для реалізації постійним у часі.

1.2. Контрзаходи проти часових атак

Одним з ефективних на даний момент заходів проти часових атак є алгоритм BlindVector, запроваджений Saarinen у 2017 році. Він є розширеним та вдосконаленим алгоритмом перемішування Фішера – Йейтса, який, як правило, використовується для ефективної випадкової перестановки коефіцієнтів вектору, розподіленого за гаусовим розподілом. BlindVector отримав покращення випадкових перемішувань. Саме це, а також постійна за часом реалізація, і дає змогу протистояти атакам побічними каналами [13, 14]. Цикли алгоритму не залежать від даних, а операції завжди виконуються, незалежно від того, замінено якесь значення чи ні.

Серед контрзаходів також можна відмітити процес відкидання зразка. Основна особливість його роботи полягає у зчитуванні з випадкових адрес додаткового кешу. Це відбувається для спотворення статистики для SCA. Після цього зайві зчитування відкидаються. Діапазон норми відкидання обрано на рівні 6,25, 12,5 та 25 % [6].

До нововведень відноситься більш ефективна конструкція гаусової вибірки (CDT) з постійним часом. Її перевагами є чітка кількість операцій пошуку-зчитування, а також застосування однакової арифметики, яка не залежить від гілки, яку було обрано згідно з CDT алгоритмом [6]. Максимальна кількість необхідних пошукових операцій оцінюється як $\lceil \log_2 N \rceil$, тому кожен виклик пробовідбірника доповнюється до найближчого ступеня двійки для того, щоб займати стільки ж тактових циклів. Побудований таким чином пробовідбірник здатен забезпечити кращу швидкодію, ніж пробовідбірник Knuth-Yao, дискретний відбірник Ziggurat'a та відбірник Бернуллі, за умови, що він виконується за усталений час.

Функції NTT та FFT отримали постійну за часом реалізацію, що також надає можливість протидії часовим атакам. Це досягається шляхом обробки всіх необхідних змінних гілки та відмови від логіки для більш тривалих арифметичних операцій. Застосування векторизації SIMD та «ледачого» скорочення (lazy reduction) дозволяє значно зменшити вплив цих контрзаходів на продуктивність [6].

Можливість множення двох кілець у домені NTT [13] забезпечується шляхом застосування поточкового модульного множення. Завдяки цьому кожен елемент обчислюється незалежно. Про усталеність у часі даних операцій свідчить їх послідовність та безумовність. Важливою особливістю є також застосування вдосконаленого NTT алгоритму Cooley-Tukey. Вдосконалення стосуються покращення продуктивності та автоматичної векторизації. Також гарантується, що умовні операції розгалуження не будуть використовуватися [6]. Автоматична векторизація виконується окремо від модульного скорочення з метою покращення продуктивності. До того ж, для модульного скорочення застосовується спрощене обмеження діапазону, що дає змогу перейти від логічних до арифметичних операцій, які є усталеними за часом в більшості випадків [6].

2. Злам схеми постквантового підпису Falcon шляхом атаки побічними каналами

Хоча алгоритми можуть бути математично обґрунтованими проти класичного або квантового криптоаналізу, їх реалізація може призвести до витоку секретної інформації через побічні канали [3]. Ці атаки знаходять кореляцію між секретними значеннями та поведінкою реалізації, як-от час виконання, енергоспоживання та електромагнітне (ЕМ) випромінювання. Серед цих атак фізичні побічні канали (наприклад, витік ЕМ) мають особливе значення, оскільки вони існують не через неправильний вибір дизайну як такого, а через фізику залежної від даних активності CMOS. Ці атаки можуть бути успішними за допомогою всього лише кількох тестів, застосованих до фізичного пристрою, і без потреби в будь-якому функціональному квантовому комп'ютері. Атаки побічними каналами також важливі для NIST, оскільки вони є критерієм для визначення кінцевого стандарту [4].

У роботі Emre Karabulut та Aydin Aysu [15] пропонується перша атака побічними каналами на Falcon. Така атака є атакою з відомим відкритим текстом, яка використовує електромагнітні вимірювання пристрою для отримання секретних ключів підпису, які потім можна

використовувати для підробки підписів у довільних повідомленнях. Запропонована атака націлена на унікальне множення з плаваючою комою в рамках швидкого перетворення Фур'є алгоритму Falcon за допомогою нової стратегії розширення та скорочення, яка отримує змінні знака, мантиси та експоненти без помилкових спрацьовувань. Потім отримані значення з плаваючою комою відображаються назад у коефіцієнти секретного ключа. Подібна атака, зокрема, не вимагає попередньої характеристики профілю потужності цільового пристрою або створення спеціальних вхідних даних. Натомість статистичні відмінності на отриманих даних достатні для успішного виконання запропонованого диференційного електромагнітного аналізу. Результати на ARM-Cortex-M4, що працює з довідковим програмним забезпеченням FALCON NIST, показують, що приблизно 10 тисяч вимірювань достатньо, щоб отримати весь ключ.

У розділі:

• Пропонується перша атака побічними каналами на FALCON. Проводиться аналіз алгоритму, виявлення вразливих обчислень, через які може статися витік інформації, і цей витік може спричинити підробку підписів у довільних повідомленнях.

• Показано, що пряма атака на цільові обчислення зазнає невдачі через помилкові спрацьовування при множенні, і представляємо нову атаку, яка може вирішувати помилкові припущення за допомогою стратегії розширення та скорочення.

• Відбувається застосування запропонованої атаки на еталонне програмне забезпечення FALCON, взяте з веб-сайту NIST, і демонструємо, що запропонована атака може отримати цілі ключі підпису за допомогою кількох тисяч вимірювань, коли FALCON працює на мікроконтролері ARM-Cortex-M4.

2.1. Попередні відомості

У цьому розділі надається довідкова інформація про алгоритм цифрового підпису Falcon і модель загрози.

А. Модель загрози (зловмисника)

Робота дотримується стандартних припущень щодо атак побічним EM-каналом, коли зловмисник має фізичний доступ до пристрою та фіксує EM-вимірювання, поки виконуються обчислення, що залежать від ключових елементів. Дві помітні переваги нашої атаки в порівнянні з деякими нещодавніми роботами з побічних каналів решітчастої криптографії полягає в тому, що не потрібно створювати спеціальні вхідні дані або інше джерело вразливості, таке як побічний канал синхронізації для вилучення секретної інформації.

Б. Схема електронного підпису Falcon [7]

Falcon – це постквантова схема підпису, заснована на решітках [4]. Falcon складається з процедур генерації ключів, підписання та перевірки підпису. В роботі детально розглядаються перші дві процедури, оскільки вони мають вирішальне значення для розуміння атаки.

В наступному алгоритмі показано процес генерації.

Алгоритм 1: Keygen(ϕ, q): вхідні дані – одиничний поліном $\phi \in \mathbb{Z}[x]$, модуль q; вихідні дані – приватний ключ sk та відкритий ключ pk.

$f, g, F, G \leftarrow \mathsf{NTRUGen}(\phi, q)$
$\mathbf{B} \leftarrow \begin{bmatrix} g & -f \\ G & -F \end{bmatrix}$
$\hat{\mathbf{B}} \leftarrow \mathsf{FFT}(\mathbf{B})$
$\mathbf{G} \leftarrow \hat{\mathbf{B}} \times \hat{\mathbf{B}}^{\star}$
$\mathsf{T} \leftarrow \mathsf{ffLDL}^{\star}(\mathbf{G})$
for each leaf leaf of T do
    leaf.value $\leftarrow \sigma / \sqrt{\text{leaf.value}}$
$\mathsf{sk} \leftarrow (\hat{\mathbf{B}}, \mathsf{T})$
$h \leftarrow g f^{-1} \bmod q$
$\mathsf{pk} \leftarrow h$
return sk, pk

Рис. 1. Алгоритм Keygen

Рис. 1 показує процедуру генерації ключа, яка створює секретний ключ sk для створення та відкритий ключ h для перевірки підписів. Вхідними даними алгоритму є параметри ϕ і q: усі операції відбуваються над монічним поліномом ϕ ступеня n, який має вигляд x^n+1 для двійкового випадку та $x^n - x^{n/2}+1$ для трійкового випадку, тоді як q є простим модулем. Цей алгоритм спочатку випадковим чином відбирає коефіцієнти поліномів f і $g \in \mathbb{Z}[x]$ із дискретного розподілу Гауса, а потім обчислює F і $G \in \mathbb{Z}[x]$, які задовольняють рівнянню NTRU $fG - gF = q \bmod \varphi$. Багаточлени f, g, F і G називають приватними елементами. Потім ці поліноми об'єднуються, пропускаються через FFT і перетворюються на повнорангову матрицю Грама G. Щоб обчислити бінарне дерево T, Falcon застосовує LDL розклад на G. Алгоритм генерації ключа повертає відкритий ключ h, який задовольняє рівняння $gf^{-1} = h$ та секретний ключ sk містить два компоненти $\hat{B}$ і T, які походять від чотирьох поліномів $f, g, F, G \in \mathbb{Z}[x]$. Таким чином, стійкість генерації ключів базується на квантовостійкій проблемі NTRU, яка спирається на складність відновлення поліномів f і g, заданих поліноміальним кільцевим елементом h. Коефіцієнти цих приватних поліномів f і g мають діапазон від -127 до +127.

Якщо дано приватний ключ sk та повідомлення m, підписувач використовує sk для підпису m таким чином [4]:

- випадковий модифікатор входу геш-функції ($salt$) r генерується рівномірно в множині $\{0,1\}^{320}$. Після цього об'єднаний рядок ($r \| m$) гешується до точки $c \in \mathbb{Z}_q[x]/(\phi)$.
- обчислюється прообраз t для c і потім подається як вхідні дані для використання алгоритмом швидкої вибірки Фур'є. На виході отримаємо два коротких поліноми $s_1, s_2 \in \mathbb{Z}[x]/(\phi)$ (у FFT представленні) такі, що $s_1 + s_2 h = c \bmod q$.
- s_2 кодується (стискається) до рядку бітів s.
- підписом є пара (r, s).

Алгоритм 2: Sign($m, sk, \lfloor \beta^2 \rfloor$): вхідні дані – повідомлення m, приватний ключ sk, межа $\lfloor \beta^2 \rfloor$; вихідні дані – підпис sig для m (рис. 2).

1: $\mathbf{r} \leftarrow \{0,1\}^{320}$ uniformly
2: $c \leftarrow \mathsf{HashToPoint}(\mathbf{r} \| \mathbf{m}, q, n)$
3: $\mathbf{t} \leftarrow \left(-\frac{1}{q}\mathsf{FFT}(c) \odot \mathsf{FFT}(F), \frac{1}{q}\mathsf{FFT}(c) \odot \mathsf{FFT}(f)\right)$
4: do
5: do
6: $\mathbf{z} \leftarrow \mathsf{ffSampling}_n(\mathbf{t}, \mathbf{T})$
7: $\mathbf{s} = (\mathbf{t} - \mathbf{z})\hat{\mathbf{B}}$
8: while $\|\mathbf{s}\|^2 > \lfloor \beta^2 \rfloor$
9: $(s_1, s_2) \leftarrow \mathsf{invFFT}(\mathbf{s})$
10: $\mathbf{s} \leftarrow \mathsf{Compress}(s_2, 8 \cdot \mathsf{sbytelen} - 328)$
11: while $(\mathbf{s} = \perp)$
12: return $\mathsf{sig} = (\mathbf{r}, \mathbf{s})$

Рис. 2. Алгоритм Sign

B. FFT над числами з плаваючою крапкою

Falcon використовує односторонній відбірник секрету і тому повинен працювати з арифметикою з плаваючою комою та FFT, а не з цілочисельною арифметикою та NTT [4]. Falcon округлює (наближує) числа з плаваючою комою, які використовуються в арифметичних операціях, подібно до стандарту IEEE 754 з плаваючою комою (подвійна точність). Це наближення представляє число з плаваючою комою 64 бітами, де MSB є знаковим бітом, наступні

11 бітів є експонентою, а решта 52 біти є мантисою. Falcon вимагає арифметики з плаваючою комою під час підписання та етапів генерації ключів (алгоритми 1 і 2).

Falcon прискорює множення кільцевих поліномів за допомогою FFT, яке працює над кільцем $\mathbb{Z}_q / \varphi(x)$, де $\varphi(x)$ є монічним скорочувальним поліномом. FFT зменшує часову складність шляхом перетворення поліномів з $\mathbb{Z}_q / \varphi(x)$ в іншу область, де поліноміальне множення стає покоефіцієнтним (скалярним) множенням. Процедура підпису перетворює гешоване повідомлення та коефіцієнти елементів закритого ключа (f, g, F, G) у числа з плаваючою комою, а потім застосовує до них перетворення домену FFT (Алгоритм 2, рядок 3). Таким чином, алгоритм FFT перетворює 8-розрядні цілі коефіцієнти елементів закритого ключа на 64-розрядні коефіцієнти з плаваючою крапкою. Алгоритм FFT застосовує операції додавання, віднімання та множення з плаваючою крапкою між коефіцієнтами вхідного полінома. Після переходу до домену FFT алгоритм підписання Falcon виконує скалярне множення з плаваючою комою між елементами приватного ключа f і F і гешованим повідомленням c.

2.2. Запропонована атака побічними каналами

У цьому розділі представлено запропоновану атаку на алгоритм цифрового підпису Falcon і пов'язані з цим проблеми. По-перше, опишемо проміжні обчислення і чому такий спосіб дозволяє відновлювати секретні ключі та підробляти підписи. Потім ми покажемо проблеми виконання атаки побічними каналами та те, як ми вирішували ці проблеми.

A. Цільова операція $FFT(c) \odot FFT(f)$ та обґрунтування

Запропонована атака спрямована на множення з плаваючою комою в рамках обчислень $FFT(c) \odot FFT(f)$ (Алгоритм 2, рядок 3). Стверджується, що можлива атака за допомогою відомого відкритого тексту на ці обчислення та що захоплення коефіцієнтів $FFT(f)$ за допомогою атаки побічними каналами дозволяє зловмиснику підписувати довільні повідомлення.

Секретні елементи Falcon складаються з поліномів f, g, F і G. Ці поліноми використовуються для обчислення компонентів закритого ключа підпису $\hat{B}$ і T (Алгоритм 2). Поліноми F і G утворюють рівняння NTRU разом з f і g; отже, якщо зловмисник знає поліноми f і g, він може обчислити F і G, вивести весь секретний ключ і успішно підписати довільні повідомлення. Оскільки відкритий ключ h також є добутком gf^{-1}, зловмиснику необхідно отримати або поліном f, або g, щоб здійснити успішну атаку.

З цією метою атака має націлюватись на операцію $FFT(c) \odot FFT(f)$ (Алгоритм 2, крок 3), де відбувається поліноміальне множення на основі FFT між гешованим повідомленням c і приватним поліноміальним елементом f. Націлювання на це обчислення за допомогою атаки побічним каналом є здійсненним, оскільки c відоме зловмиснику, і, отже, секрет f можна припустити та перевірити через витік.

На рис. 3 показано деталі цільового множення на основі FFT між гешованим повідомленням c і приватним елементом f:

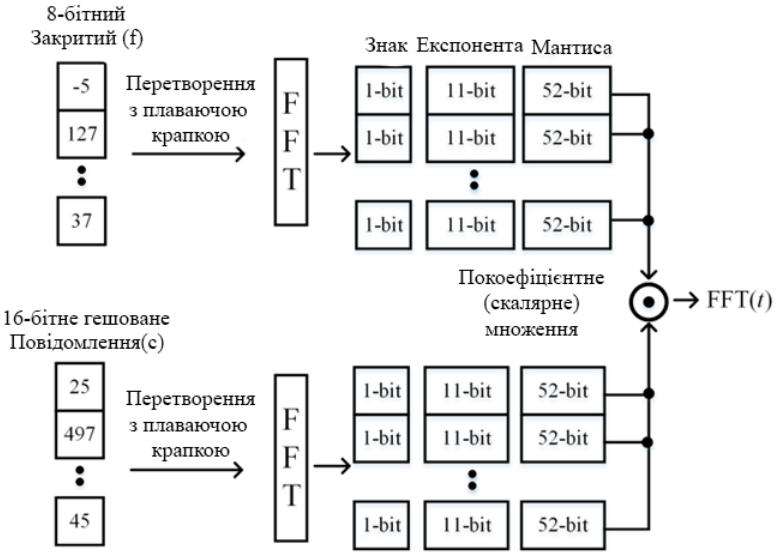

Рис. 3. Множення приватного елемента f і полінома гешованого повідомлення c.
Атака спрямована на скалярні множення з плаваючою крапкою в домені FFT

Еталонна реалізація Falcon спочатку перетворює цілі коефіцієнти поліномів c і f у число з плаваючою крапкою. Потім Falcon застосовує перетворення FFT над цими коефіцієнтами з плаваючою крапкою. Після перетворення FFT Falcon виконує скалярне множення з плаваючою крапкою між 64-бітними коефіцієнтами $FFT(c)$ та $FFT(f)$. Оскільки рівномірно випадкова сіль r і повідомлення m є загальнодоступними, $FFT(c)$ може бути обчислений зловмисником. Таким чином, запропонована атака зосереджена на множенні з плаваючою крапкою між коефіцієнтами відомого $FFT(c)$ і секретного $FFT(f)$. Якщо зловмисник успішно виділяє коефіцієнти полінома $FFT(f)$, він може відновити приватний елемент f, оскільки функція FFT Falcon є оборотною та однозначною.

Б. Отримання $FFT(f)$ шляхом підходу «Розділяй-і-володарюй» [15]

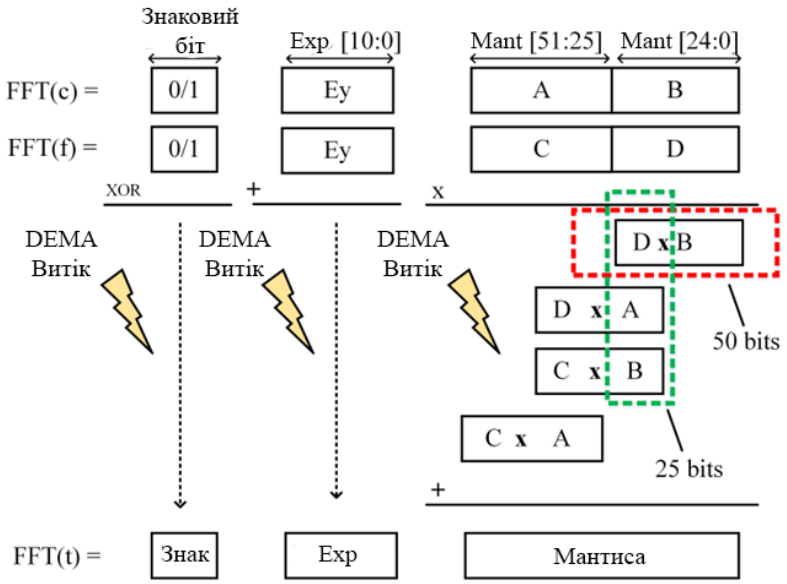

Рис. 4. Пропонована атака на множення FALCON з плаваючою крапкою. Націлювання на множення (показано червоними пунктирними лініями) створює хибні спрацьовування, а націлювання на проміжні додавання (показано зеленими пунктирними лініями) усуває їх

На рис. 4 показано кроки множення з плаваючою крапкою для двох коефіцієнтів. Операція множення приймає два 64-бітні коефіцієнти та генерує 64-бітний вихід. Операція складається з трьох частин: множення мантиси, додавання експоненти та обчислення знакового біта. Множення мантиси має чотири кроки. Перший об'єднує 52-бітну мантису вхідних коефіцієнтів з одним бітом «1», щоб зробити його старшим бітом (MSB). Другий розділяє біти мантиси на 28 біт вищого порядку та 25 біт нижчого порядку. Третій застосовує звичайне множення цілих чисел на розділені біти мантиси двох коефіцієнтів. Така послідовність кроків генерує 106-розрядні добутки. Оскільки кінцевий розмір мантиси має становити 52 біти, нижні біти називаються невикористаними, закріпленими бітами. Таким чином, четвертий крок полягає в тому, щоб округлити добуток множення до 52 бітів, видаливши ці закріплені біти.

Falcon застосовує додавання експоненти до 11-бітових експонент вхідних коефіцієнтів. Додавання експоненти також отримує додатковий біт переносу з результату множення мантиси. Останньою операцією є обчислення знакового біта, який є операцією XOR між бітами MSB вхідних коефіцієнтів.

Для атаки множення з плаваючою крапкою у Falcon пропонується використання стратегії «розділяй-і-володарюй». Запропонований метод атаки окремо відновлює біти мантиси, експоненти та знака та поєднує їх для отримання коефіцієнтів $FFT(f)$. Без такої стратегії проста диференціальна атака потребувала б створення масивних таблиць із 2^{64} записами для кожного коефіцієнта. На рис. 5 показано отриманий ЕМ-трафік цільового множення з плаваючою крапкою між двома коефіцієнтами [15]:

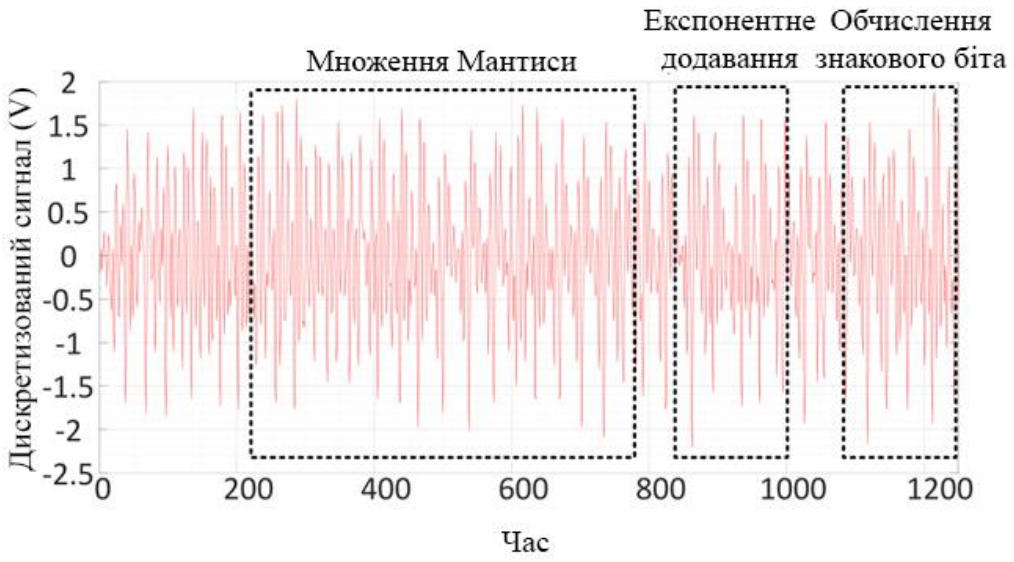

Рис. 5. Приклад даних (сліду) електромагнітних вимірювань з експерименту, що показує відповідні обчислення мантиси, експоненти та знаку

Червона лінія позначає електромагнітний сигнал, а чорні штрихові лінії вказують на те, протягом якого часу виконуються операції з мантисою, експонентою та знаком.

Ключовою проблемою в досягненні цієї атаки є усунення помилкових спрацьовувань, які виникають під час множення мантиси. Дійсно, відомо, що множення дає хибні позитивні результати, оскільки результати корельовані, тобто подібні коефіцієнти дадуть подібний результат множення. Наприклад, коефіцієнт $FFT(f)$ «1» і «2» генеруватиме однакові значення, зміщені на одну двійкову цифру. Це вирішується новою технікою розширення та скорочення [15].

B. «Розширення та скорочення» для видалення хибних спрацьовувань

Ми вирішуємо ключову проблему усунення хибно-позитивних результатів множення за допомогою нової техніки розширення та скорочення. Суть цієї техніки полягає у використанні найкращих припущень, отриманих під час атаки на множення (фаза розширення) та оцінки їх правильності шляхом атаки на наступну операцію (фаза скорочення), яка є додаванням проміжних результатів (рис. 2).

Хоча поліноміальні коефіцієнти f визначаються як цілі числа в діапазоні від -127 до 127, вони перетворюються на числа з плаваючою крапкою, а потім передаються в FFT домен. Оскільки алгоритм FFT змішує та розсіює всі вхідні коефіцієнти та оскільки FFT виконує арифметику з плаваючою крапкою, коефіцієнти домену FFT мають діапазон $[0, 2^{64}]$, навіть якщо вхідні дані мають діапазон $[-127,127]$.

Описана функція «Розширення та скорочення» налаштована під реалізацію арифметики з плаваючою крапкою Falcon. Щоб отримати частину з мантисою, ми спочатку виконується атака 25 біт молодшого порядку множення – на рис. 4 показано 25 біт молодшого порядку як B і D, де B відоме, а D – секретне значення. Запропонована диференціальна атака на множення $D \times B$ виконується за звичайними кроками: створення гіпотетичних припущень щодо секретних 25 біт нижчого порядку (D), обчислення очікуваної активності перемикання для кожного множення (з використанням ваги Хеммінга) і перевірка кореляції між ними та відповідними електромагнітними вимірюваннями. Та сама процедура повторюється і для $D \times A$, де A відоме, а D є секретним значенням. Це розширена фаза атаки, яка, як очікується, призведе до помилкових спрацьовувань.

Другим кроком у атаці є фаза скорочення, де ціллю є додавання $D \times B$ і $D \times A$ для скорочення помилкових позитивних значень і відновлення 25 біт секретної мантиси. На відміну від множення, додавання не призведе до хибних позитивних результатів: наприклад, однакові коефіцієнти «1» проти «2» генерують результати з різними вагами Хеммінга на основі інших вхідних даних додавання. За наявності достатньої кількості тестів секретні коефіцієнти «1» проти «2» (та інші випадки, які дають хибно-позитивний результат множення) можна відрізнити один від одного [15].

Для отримання D при операції додавання також застосовується вищезгаданий порядок дій. Біти вищого порядку множення мантиси виконують ті самі кроки множення та додавання. Тому до старших 27 бітів секретних коефіцієнтів застосовується та сама техніка розширення та скорочення. Зауважте, що ми не обходимо перше множення й безпосередньо атакуємо операцію додавання, оскільки розташування бітів добутку додавання $D \times B$ і $D \times A$ не узгоджуються одне з одним, що призводить до зниження успіху атаки.

Запропонована атака застосовує ту саму процедуру диференціальної ЕМ-атаки (DEMA) для вилучення бітів експоненти та бітів знака. На рис. 4 показано, що в реалізації з плаваючою крапкою відбувається додавання двох експонент поліномів $FFT(c)$ та $FFT(f)$ і застосовує операцію XOR до знакових бітів. Комбінована версія окремо відновлених бітів мантиси, експоненти та знака представляє один повний коефіцієнт [15].

2.3. Результати оцінювання

У дослідженні було використано загальнодоступне довідкове програмне забезпечення Falcon Round 3, яке міститься в пакеті подання до стандартизації NIST. Код було зкомпільовано за допомогою компілятора gcc-arm-none eabi-4_8-2014q1 і з міткою (прапором) -O0, а потім перенесено згенерований виконуваний файл на ARM-Cortex M4. Процесор має тактову частоту 168 МГц, а вимірювання виконуються за допомогою EM Probe LS (низька чутливість) RISC-EMP430LS, який є датчиком ближнього поля для вимірювання до 1 ГГц із чутливістю 20 мВ/1 Т@1 МГц. Шум вимірювань електромагнітного датчика зменшується за допомогою дросельної котушки та відбирається за допомогою осцилографа PicoScope 3206D зі швидкістю 500 Мс/с.

Для запропонованої диференціальної атаки побічним каналом ми використовуємо розрізнювач на основі коефіцієнта кореляції Пірсона на вагових моделях Хеммінга. Цей тест має на меті диференціювати популяції через їх коваріацію, тобто шляхом перевірки того, чи відхилення від середнього відбуваються подібним чином. Кореляційний слід $r_{i,j}$ для припущення i визначається як [15]:

$$r_{i,j} = \frac{\sum_{d=1}^{D}[(h_{d,i} - \bar{h}_i)(t_{d,j} - \bar{t}_j)]}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \bar{h}_i)^2 \sum_{d=1}^{D}(t_{d,j} - \bar{t}_j)^2}} \quad (1)$$

де D – це кількість слідів (трейсів), кожен з яких має T точок даних, $t_{d,j}$ – це електромагнітний слід з $0 < d \leq D$ і $0 < j \leq T$, $\bar{t}_j$ – середнє значення електромагнітного сліду, $h_{d,i}$ – оцінка витоку в сліді d для припущеного значення i, а $\bar{h}_i$ є середнім значенням цієї оцінки. Результат $r_{i,j}$ повертає кореляційний слід (трейс) зі значеннями в діапазоні $[-1,1]$, який є оцінкою лінійного зв'язку між припущеннями r_i та електромагнітними вимірюванням для кожного припущення i та часу j. Цей слід відображає значення та інформацію про час диференціального електромагнітного витоку [15].

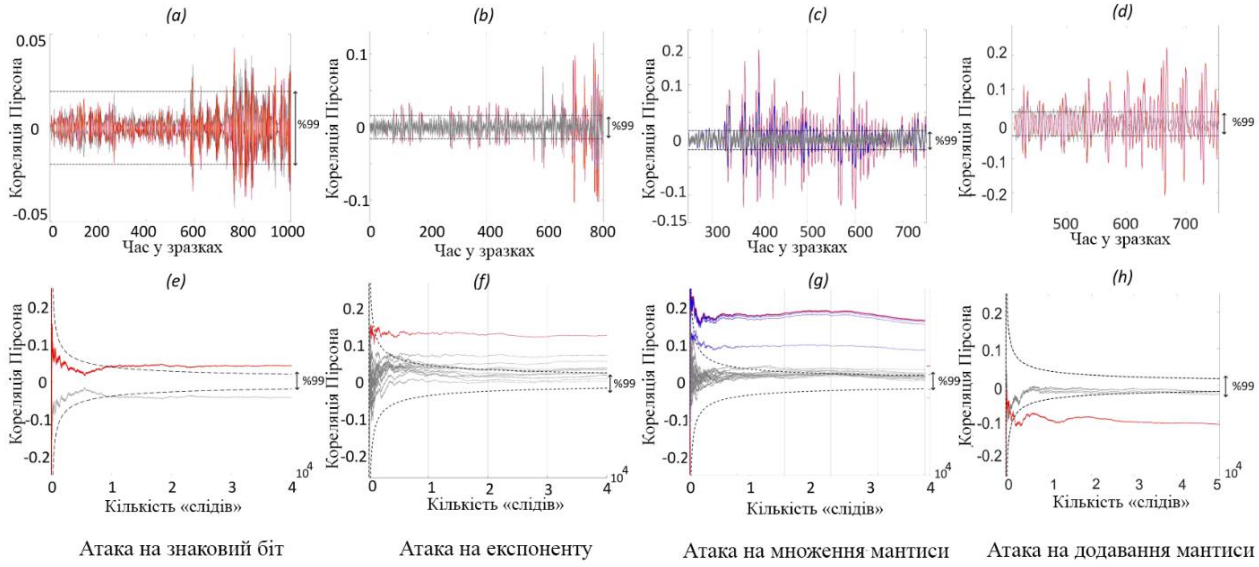

Рис. 6. Результати запропонованої атаки на коефіцієнт з плаваючою крапкою під час підписання за допомогою Falcon. Правильні припущення виділені жирним червоним шрифтом, значні помилкові припущення синім, а пунктирні лінії позначають довірчий інтервал 99,99 %. Attack повертає правильні припущення для (*a*) знака, (*b*) експоненти, (*c*), множення мантиси (*c*) і додавання (*d*), підтверджуючи наш підхід. Для цього коефіцієнта правильні значення можна отримати з достовірністю понад 99,99 % менш ніж за 10 тисяч вимірювань

На рис. 6 показано результати запропонованих атак, і це підтверджує, що атаки працюють на практиці та можуть зламати Falcon. Атака виконується на коефіцієнт 0xC06017BC8036b580, що означає, що правильний знаковий біт – 0x1, експонента – 0x406, а біти мантиси – 0x017bc8036b580 (старші біти – 0x00BDE40, а нижчі – 0X36b580). Часові графіки кореляції на рис. 6, *a* – *d* відповідно кількісно визначають, що кореляційні сліди ($r_{i,j}$) й для правильних і хибно-позитивних припущень для знака, експоненти та мантиси перетинають 99,99 % довірчий інтервал, тоді як ті, що знаходяться в межах є істинно негативними, тобто ми не маємо помилково негативного випадку. Графік показує лише до найкращих

21 припущення для частини мантиси для візуальної ясності – атака оцінює до 2^{27} припущень для бітів мантиси вищого порядку та 2^{25} припущень для бітів нижчого порядку.

Рис. 6, *c*, *d* підтверджують нашу гіпотезу про запропоновану атаку. Дійсно, коли атака виконується на множення бітів мантиси, результати на рисунку 6, *c* показують, що значні помилкові спрацьовування відбуваються на першому кроці атаки, коли вона спрямована на множення мантиси. Результат кореляції п'яти найкращих припущень, які включають правильне припущення та чотири помилкові позитивні результати, насправді є абсолютно однаковими (дещо відрізняються на рисунку для наочності). Однак результати на рис.6, *d* підтверджують, що за допомогою подібної стратегії розширення та скорочення всі хибні спрацьовування усуваються, коли атака потім відстежує припущення на попередньому кроці, зосереджуючись на проміжних доповненнях [15].

На рис. 6, *e* – *h* зображено еволюцію кореляції, взяту за вибірку часу з найменшим витоком на рис. 6, *a* – *d* відповідно. Ці графіки вимірюють кількість слідів, необхідних для досягнення статистично значущої (99,99 %) кореляції. Витік правильних припущень стає статистично значущим лише з тисячею вимірювань під час атаки на експоненту та додавання мантиси, тоді як інші припущення зникають [15].

Найскладнішою частиною атаки, з точки зору кількості необхідних вимірювань, є вилучення знакового біта, і потрібно близько кількох сотень вимірювань, щоб отримати правильне значення, і приблизно 9 тисяч вимірювань, щоб зробити витік статистично значущим для цього прикладу. Загалом, вимірювання для всіх коефіцієнтів можна впевнено отримати менш ніж за 10 тисяч вимірювань. Зауважте, що витік знакових бітів є симетричним для припущень позитивного та негативного знаку. Це справді очікувано і не створює проблем для атаки, оскільки правильне припущення постійно має позитивну кореляцію в максимальній точці витоку (тобто червона лінія на рис. 6, *e* виглядає однаково як для негативних, так і для позитивних знаків) [15].

Було здійснено запропоновану атаку на Falcon-512, але така сама атака може бути застосована до іншого набору параметрів, тобто Falcon-1024, оскільки Falcon використовує однакові арифметичні реалізації з плаваючою крапкою для обох наборів параметрів.

2.4. Особливості

А. Обмеження подібної атаки

В роботі запропоновано загальну атаку, яка працює без профілювання цільового пристрою шляхом (пере)конфігурації секретного ключа, і це все ще практично. Можна розширити атаку за допомогою шаблонів або методів профілювання на основі машинного навчання і використання кращого вимірювального обладнання.

Б. Можливі контрзаходи

Найпопулярнішими методами захисту побічних каналів є приховування та маскування. У той час як приховування має на меті зробити енергоспоживання постійним, маскування має на меті рандомізацію проміжних значень, оброблених реалізацією. Хоча SABRE, ще один фіналіст постквантової сигнатури NIST, нещодавно запропонував замасковану реалізацію, вона ще не існує для Falcon.

В. NTT проти FFT – перспектива щодо побічних каналів

Ґрунтуючись на аналізі, можна стверджувати, що FFT, ймовірно, має нижчий рівень витоку потужності/ЕМ випромінювання за побічними каналами порівняно з NTT. Хоча для подібної атаки на FFT потрібно близько 10 тисяч трас, NTT виявилася вразливою навіть з одним слідом. Ймовірно, це пов'язано з великою кількістю нелінійності. У той час як NTT застосовує модульне скорочення простих p, цього не відбувається у FFT. Таким чином, атака розрізнить і усуне неправильні припущення в NTT значно простіше і швидше.

У цьому розділі продемонстровано першу атаку побічним каналом на постквантового фіналіста NIST Falcon. Результати підтвердили, що подібні атаки дійсно працюють на практиці з кількома тисячами вимірювань і без квантового комп'ютера. Таким чином, робота

обґрунтовує необхідність ефективних заходів протидії таким атакам і врахування відповідних накладних витрат у показниках продуктивності апаратного/програмного забезпечення [15].

3. Атака помилками на Falcon – BEARZ

BEARZ [6] представляє собою атаку помилками на схему електронного підпису Falcon. Дана атака була вперше запропонована Sarah McCarthy та ін. у 2019 році. Заснована вона на вилученні базису через переривання рекурсії або обнулення. Викликаючи помилки в роботі схеми, атака добуває закриту ключову інформацію. Моделлю зловмисника передбачено можливості пропуску команд та обнулення змінних.

3.1. Рекурсія алгоритму Falcon

Оскільки алгоритм вибірки для Falcon є рекурсивною формою GPV відбірника, атака буде спиратися на переривання рекурсивного виклику на початку. Як було сказано вище, Falcon [7, 12] має два рекурсивні виклики на верхньому рівні алгоритму ffSampling. Наприклад, задано вектор (t_0, t_1), алгоритм застосовується спочатку для t_1, а потім для t_0. Кожен елемент (t_0, t_1) безперервно ділиться на два вектори довжини $n = n/2$ поки n не буде дорівнювати 2. Після цього відбираються коефіцієнти гаусового розподілу для отримання вектору вибірки (z_0, z_1). Це спричиняє присутність на верхньому рівні алгоритму двох рекурсивних гілок. Для вектору довжини n, перші $n/2$ значень будуть представлені реальними коефіцієнтами, а другі $n/2$ – уявними [6].

Успішна атака потребуватиме переривання рекурсивного виклику у необхідній точці таким чином, щоб тільки $m-n$ місць були б заповнені. Але через характер функцій FFT така операція є неможливою.

3.2. FFT: злиття та розділення

Функції злиття та розділення для схеми Falcon виконуються у домені FFT. Це означає, що нульові вхідні дані не представлені нулем, що в свою чергу є проблемою для атаки помилками [6]. Тому після отримання вектору решітки із гаусового зразка, до нього застосовується зворотня FFT функція FFT^{-1}. Це робиться з метою переконатися, що підпис не знаходиться у FFT домені. Коли всі перераховані кроки виконані, може бути використана та сама післяобробка, що і для DLP-атаки.

3.3. Атаки переривання рекурсії

Атаки переривання рекурсії класифікуються в залежності від місця, в якому відбувається переривання. Усі вони призводять до однакового вихідного формату вектору z; перші $(2n-m)$ коефіцієнтів $z = (z_0, z_1)$ примусово задаються нулями.

Переривання другої рекурсії (для $m = n$)

Атака здійснюється шляхом переривання в кінці виклику алгоритму вибірки, після першого рекурсивного виклику [6]. Якщо атака виявиться успішною, то z_1 заповниться вибраними коефіцієнтами, а z_0 залишиться повністю нульовим. Такий тип атаки може бути виконаний також для $m \le n$, з метою обнулення перших n коефіцієнтів.

Обнулення або атака пропуску (для $m \le n$)

Існує два варіанти подібної атаки. Перший: при передостанньому злитті встановити необхідні з вихідних коефіцієнтів в нуль шляхом пропуску операцій або подальшого обнулення необхідних коефіцієнтів [6]. Операції, які необхідно пропустити: $f[(u << 1) + 0] = t\_re$ та $f[(u << 1) + 1] = t\_re$ з коду (Prest та ін., 2017)($merge\_fft()$ функція у $falcon\_fft.c$) [4, 9].

Другий: встановити необхідну кількість початкових коефіцієнтів z_1 в нуль перед обчисленням відповідного вектору решітки, тобто перезаписати вихід z_1 відбірника [6].

Переривання посеред рекурсії (для $m \le n$)

Даний тип потребує попереднього обчислення (одноразового), зате дає змогу застосовувати помилку на стадії одновимірного гаусового відбірника [9, 10, 16]. Це є сприятливою умовою, оскільки передбачає простоту фізичного вбудовування.

Якщо необхідно прирівняти до нуля ліву половину z_1 (тобто $m = n/2$), то вектор лівої сторони (LHS) при останньому виклику $merge\_fft()$ має містити у своїй першій половині нульові значення, а у векторі правої сторони (RHS) перша половина коефіцієнтів має дорівнювати його другій половині. Кожен дійсний коефіцієнт вектору z_1 генерується як $z_1[2u] = f_0[u] + (f_1[u] - f_1[u + n/4])$ та $z_1[2u+1] = f_0[u] - (f_1[u] - f_1[u + n/4])$, де n – розмірність вектору вищого рівня, а $u \in \{0, ..., n/4 - 1\}$. Для обнулення $z_1[2u]$ та $z_1[2u+1]$, можна встановити $f_0[u] = 0$ та $f_1[u] = f_1[u + n/4]$ для кожного $u \in \{0, ..., n/4 - 1\}$. Наприклад, для $n = 512$, перші $n/4 = 128$ коефіцієнтів 256-мірного вектору LHS встановлюються рівними нулю, а перші 128 коефіцієнтів 256-мірного вектору RHS встановлюються рівними другим 128 коефіцієнтам цього ж вектору [6].

256-розмірній RHS вектор повинен містити в першій половині значення, рівні значенням другої половини. Для забезпечення цього 128-мірний вектор LHS має містити в собі дійсні значення, що дорівнювали б уявним значенням, а 128-розмірний RHS вектор має бути рівним нулю.

Це обумовлюється тим, що нам необхідно, щоб $f[2u] = f[2u + n/2]$ та $f[2u+1] = f[2u+1+n/2]$ були рівними для кожної ітерації u. З погляду функції $merge\_fft()$ це означає: $f_0[u] + (f_1[u] - f_1[u + n/4]) = f_0[u + n/4] + (f_1[u] + f_1[u + n/4])$ та $f_0[u] - (f_1[u] - f_1[u + n/4]) = f_0[u + n/4] - (f_1[u] + f_1[u + n/4])$. Щоб відповідати умовам рівнянь, можна встановити $f_0[u] = f_0[u + n/4]$ – так, що перша половина коефіцієнтів дорівнюватиме другій половині, та $f_1[u] = f_1[u + n/4] = 0$. Таким чином, рівняння будуть просто залежати від вектору подачі LHS [6].

Кожну нижню гілку для RHS можна встановити рівною нулю. Якщо взяти LHS 128-розмірний вектор: 64-мірний вектор LHS повинен мати рівні дійсні та уявні значення, а RHS має бути нульовим. Будь-який вектор, рівний нулю, повинен мати обидва вектори подачі, рівними нулю, тому гілки нижче цієї можуть бути обнулені [16]. Даний метод може бути застосований для будь-якого m, що відповідає умові $m = 2^k$, де $k \in \mathbb{Z}$.

3.4. Обробка після нападу

Після проведення попередніх кроків та отримання недійсного підпису, останнім кроком атаки є відновлення таємного базису з даного підпису [6, 16]. Припустимо, мається $2n - m$ перших коефіцієнтів вектору (z_0, z_1) рівних нулю. Тоді підпис Falcon s_2 буде обчислюватись як

$$s = t - z\hat{B}, \tag{2}$$

де $\hat{B}$ матрицею базису у FFT домені.

Але в нашому випадку важливою є лише друга половина s, тобто s_2:

$$s_2 = t_1 - z\left(\frac{-A(f)}{-A(F)}\right). \tag{3}$$

Оскільки t_1 встановлено в 0:

$$s_2 = -z\left(\frac{-A(f)}{-A(F)}\right). \quad (4)$$

До того ж перші $m-n$ коефіцієнтів z нульові, отже:

$$s_2 = -z_1(-A(F)). \quad (5)$$

А оскільки відомо, що деякі з коефіцієнтів z_1 нульові, отримуємо:

$$s_2 = (z_1[m-1]x^{n-m}F + ... + z_1[0]x^{n-1}F). \quad (6)$$

Таким чином, вдається отримати підрешітку решітки, яка була згенерована з використанням F. А отже, маючи декілька недійсних підписів, є можливим знаходження решітки, породженої F (F – короткий вектор у даній решітці). Далі за допомогою алгоритму BKZ можна знайти цей короткий вектор [6]. Знаючи F, можна отримати G, f і g з відкритого ключа h, і тим самим знайти таємний базис NTRU решітки.

3.5. Модель помилки та контрзаходи

У моделі помилки передбачено проведення аналізу побічних каналів [6]. Шляхом аналізу може бути виявлено вікно між першим і другим рекурсивним викликом. В межах цього вікна алгоритм може бути перервано. Атака обнулення може бути застосована у момент зберігання вектору в оперативній пам'яті, шляхом обнулення під час цього необхідних бітів [16]. Альтернативним варіантом може бути пропуск рядків коду. Даний тип атаки можна організувати за допомогою перепадів тактової частоти процесора [16].

Для протистояння BEARZ [6] атаці існують певні контрзаходи. Наприклад, подвійне обчислення підпису – один з найпростіших методів виявлення атак помилками. Таким чином, відразу після підписання повідомлення підписувач має змогу переконатися в тому, що на обладнання не здійснювались атаки помилками в момент підписання. Ще одним ефективним методом виявлення BEARZ атаки є перевірка того, що вибраний вектор не йде до нуля в певній точці вздовж своєї довжини в кінці ffSampler алгоритму.

3.6. Модель помилки та контрзаходи

Даний пункт демонструє порівняльні характеристики алгоритму Falcon в чистій реалізації та реалізації з застосованими контрзаходами.

Для порівняння були обрані два набори параметрів, які застосовуються для схеми ЕП Falcon [7, 12]. Дані параметри показані в табл. 1.

Таблиця 1

Набори параметрів для тестування

Набір параметрів	Розмірність (N)	Модуль (q)
Набір 1	512	12289
Набір 2	1024	

Порівняння реалізацій з різними контрзаходами наводиться в табл. 2. Тестування проводилося з використанням CPU AMD Ryzen 7 3750H @ 2,3 – 4,0 ГГц.

Таблиця 2

Результати аналізу продуктивності

Реалізація	Набір параметрів	% погіршення
Чиста реалізація (Thomas Prest та ін.)	Набір 1	-
	Набір 2	-
+ постійне за часом NTT	Набір 1	-
	Набір 2	-
+ перевірка після підпису	Набір 1	5,5
	Набір 2	5,5
+ нульова перевірка	Набір 1	34
	Набір 2	16,5
+ відкидання зразка (6,25%)	Набір 1	14
	Набір 2	8,5
+ перемішування Фішера-Йейтса	Набір 1	21
	Набір 2	3
+ BlindVector	Набір 1	12
	Набір 2	52
Рекомендований набір контрзаходів: + перевірка після підпису + відкидання зразка (6,25%) + перемішування Фішера-Йейтса	Набір 1	4,6
	Набір 2	4,4

Для більшої наочності результати представлено у вигляді діаграми:

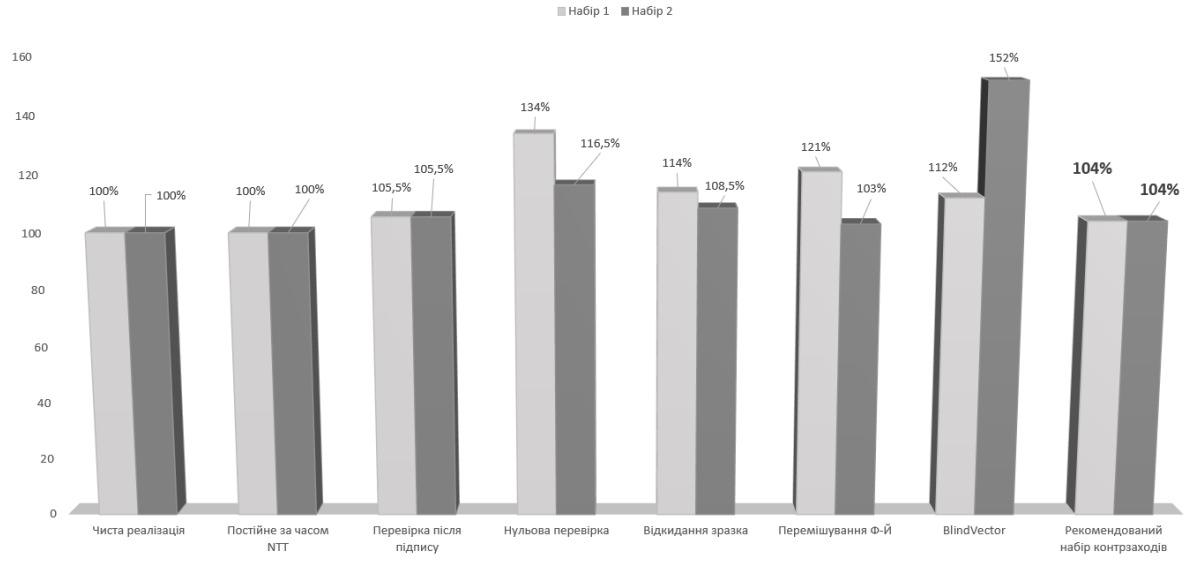

Рис. 7. Вплив контрзаходів на швидкодію

Як можна спостерігати в табл. 1 та на рис. 7, контрзаходи впливають на швидкодію Falcon незначним чином. Рекомендований набір контрзаходів погіршує продуктивність менш ніж на 5 %. Такий хороший результат зумовлений ефективністю процесів вибірки та перевірки.

Для перевірки контрзаходу проти атак помилками було 100 разів проведено процедуру підписання. Для набору 1 швидкість була в діапазоні 50 операцій на секунду, а для набору 2 – 20 операцій на секунду. Отримані дані свідчать про його ефективність проти часових атак. Також, можна додати, що контрзахід нульової перевірки виявляє атаку зі 100 % успіхом. Даний захід можна рекомендувати як мінімальний і достатній контрзахід.

Висновки

1. Після проведення аналізу вразливостей та стійкості алгоритму Falcon проти спеціальних атак можна сказати, що недоліком даного алгоритму є мала кількість досліджень стійкості проти спеціальних атак. Було показано, що певні математичні компоненти, які використовуються в алгоритмі ЕП Falcon, є вразливими до атак, заснованих на аналізі часових показників.

2. Також, що відбірник секретної інформації заснований на решітках, який застосовується у алгоритмі Falcon, настільки ж вразливий до атак помилками, як і відбірник, що використовується в альтернативних схемах підпису. Через це при стандартизації чи впровадженні слід розглядати можливість фізичних атак.

3. Вразливими до спеціальних атак є вибірка Гауса, теоретичне перетворення числа (NTT) та функція гешування HashtoPoint.

4. Аналіз можливих контрзаходів для протидії спеціальним атакам, показав, що вони впливають на швидкодію Falcon незначним чином. Рекомендований набір контрзаходів погіршує продуктивність менш ніж на 5 %. Такий перспективний результат зумовлений ефективністю процесів вибірки та перевірки.

5. Отже, незважаючи на те, що відбірник схеми Falcon все ж є вразливим до атак з помилками, ефективність компонентів та математики даного алгоритму електронного підпису сприяє тому, що він здатен конкурувати з іншими схемами, навіть з контрзаходами проти цих атак.

6. Розглянуто атаку побічними каналами на Falcon. Така атака є атакою з відомим відкритим текстом, яка використовує електромагнітні вимірювання пристрою для отримання секретних ключів підпису, які потім можна використовувати для підробки підписів у довільних повідомленнях. Запропонована атака націлена на унікальне множення з плаваючою комою в рамках швидкого перетворення Фур'є алгоритму Falcon за допомогою нової стратегії розширення та скорочення, яка отримує змінні знака, мантиси та експоненти без помилкових спрацьовувань.

7. Отримані значення з плаваючою комою відображаються назад у коефіцієнти секретного ключа. Подібна атака, зокрема, не вимагає попередньої характеристики профілю потужності цільового пристрою або створення спеціальних вхідних даних. Натомість статистичні відмінності на отриманих даних достатні для успішного виконання запропонованого диференційного електромагнітного аналізу. Результати на ARM-Cortex-M4, що працює з довідковим програмним забезпеченням FALCON NIST, показують, що приблизно 10 тисяч вимірювань достатньо, щоб отримати весь ключ.

8. Запропоновано загальну атаку, яка працює без профілювання цільового пристрою шляхом (пере)конфігурації секретного ключа, і це все ще практично. Є можливість розширити атаку за допомогою шаблонів або методів профілювання на основі машинного навчання і використання кращого вимірювального обладнання.

9. Основними методами протидії подібним атакам можна вважати приховування та маскування. Які, наприклад, реалізовані у SABRE [4], ще одному фіналісті конкурсу NIST.

Список літератури:

1. P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer // SIAM Review, vol. 41, no. 2, pp. 303– 332, 1999.

2. J. Proos et al. Shor's discrete logarithm quantum algorithm for elliptic curves // Quantum Info. Comput., vol. 3, no. 4, pp. 317–344, Jul. 2003.

3. P. Kocher et al. Differential power analysis // Advances in Cryptology – CRYPTO' 99, 1999, pp. 388–397.
4. Post-Quantum Cryptography. Round 3 Submissions. 2020. URL: https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions.
5. L´eo Ducas, Vadim Lyubashevsky and Thomas Prest. Efficient Identity-Based Encryption over NTRU Lattices. 2014. URL: https://eprint.iacr.org/2014/794.pdf.
6. Sarah McCarthy, James Howea, Neil Smythb, Séamus Brannigan, and Máire O'Neill. BEARZ Attack FALCON: Implementation Attacks with Countermeasures on the FALCON signature scheme. 2019. URL: https://eprint.iacr.org/2019/478.pdf.
7. Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang. Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU Specifications v1.2. 2020. URL: https://falcon-sign.info/falcon.pdf.
8. Verbauwhede I., Karaklajic D., and Schmidt J.-M. The Fault Attack Jungle – A Classification Model to Guide You. 2011. URL: https://www.esat.kuleuven.be/cosic/publications/article-2046.pdf.
9. J. Ahrens and U. Dieter. Extension of forsythe's method for random sampling from the normal distribution. 1973. URL: https://www.ams.org/journals/mcom/1973-27-124/S0025-5718-1973-0329190-8/S0025-5718-1973-0329190-8.pdf.
10. Thomas Prest. Gaussian Sampling in Lattice-Based Cryptography. 2015. URL: https://tel.archives-ouvertes.fr/tel-01245066v2/document.
11. Patrick Longa and Michael Naehrig. Speeding up the Number Theoretic Transform for Faster Ideal Lattice-Based Cryptography. 2016. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/05/RLWE-1.pdf.
12. Офіційний сайт ЕП Falcon. URL: https://falcon-sign.info.
13. Hodgers P., Regazzoni F., Gilmore R., Moore C., and Oder T. State-of-the-art in physical side-channel attacks and resistant technologies. 2016. URL: https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5a63fd691&appId=PPGMS.
14. Robert Primas. Side-Channel Attacks on Efficient Lattice-Based Encryption. 2017. URL: https://diglib.tugraz.at/download.php?id=5a1def5f2e7fa&location=browse.
15. E. Karabulut and A. Aysu. Falcon down: Breaking falcon post-quantum signature scheme through side-channel attacks // 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 691-696, 2021.
16. Дерев'янко Я.А., Горбенко І.Д. Вимоги та результати оцінки захищеності перспективного електронного підпису від спеціальних атак. 2020. URL: http://www.viti.edu.ua/files/zbk/2020/c_2020.pdf.

Надійшла до редколегії 10.09.2022

Відомості про авторів:

Дерев'янко Ярослав Андрійович – студент кафедри безпеки інформаційних систем і технологій, факультету комп'ютерних наук; Харківський національний університет імені В.Н. Каразіна; 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