**Вимоги та результати оцінки захищеності перспективного ЕЦП від спеціальних атак.**

**Дерев’янко Я.А., Горбенко І.Д., …**

**Анотація:**

Загроза з боку квантових комп’ютерів спричинила необхідність до якнайшвидшого дослідження постквантової криптографії. Про це наголошує Національний інститут стандартів і технологій США (NIST) у своєму нещодавньому заклику до стандартизації цього напрямку. Дане дослідження надає аналіз спеціальних атак на перспективний алгоритм Falcon.

Falcon – алгоритм електронно-цифрового підпису, що заснований на математиці алгебраїчних решіток. Він є одним з кандидатів для NIST з гарними показники швидкодії. Мінусом даного алгоритму є мала кількість досліджень стійкості проти спеціальних атак. У цьому дослідженні показується, 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 - GPV.

У даному дослідженні також проводиться порівняння швидкодії алгоритму Falcon у чистій реалізації і з урахуванням застосованих заходів, які допомагають запобігти атакам помилками та часовим атакам на схему. Воно показує, що застосування контрзаходів спричиняє лише до 30% погіршення продуктивності на етапі генерації ключових пар і лише до 5% на етапі підписання.

**1. Вступ**

Виникнення нової ери криптографії – так званої пост-квантової або квантово безпечної – яка була б здатна протистояти атакам квантового обчислення, пов’язують з відкриттям алгоритму Шора у 1999 році. Цей алгоритм мав потенційну можливість зламати всі актуальні на той час схеми підписів.

Як одне з ефективних постквантових рішень була запропонована криптографія на основі алгебраїчних решіток. Алгебраїчні решітки мають змогу протистояти навіть найсерйознішим криптоаналітичним атакам, оскільки задачі, засновані на припущеннях щодо решітки, забезпечують достатній показник захищеності від середніх до найгірших випадків (Ajtai and Dwork, 1997). До переваг роботи з решітками можна також віднести гомоморфне шифрування та шифрування на основі ідентичності. Підписам на решітках притаманні порівняно невеликі розміри ключів, підвищена швидкодія і, найголовніше – захищеність від квантових атак. Falcon є одним з трьох підписів на основі решітки серед учасників 2 раунду стандартизації NIST США.

Незважаючи на значні переваги, схеми на основі алгебраїчних решіток мають і недоліки.. Основною проблемою є те, що постквантові схеми так само вразливі до атак по бічних каналах, як і актуальні криптопримітиви. Саме тому слід враховувати можливі атаки та розглядати необхідні контрзаходи щодо них. На даний момент актуальними проти алгебраїчних решіток є часові атаки та атаки помилками. Тому це дослідження зосереджено саме на аналізі подібних атак на схему ЕЦП Falcon та можливих контрзаходах, які змогли б забезпечити захист від них.

Головним зауваженням щодо Falcon, як до кандидата у другий раунд стандартизації, була непевність щодо його захищеності від атак по бічних каналах. Це дослідження розглядає існуючі атаки на реалізацію та аналізує швидкодію схеми підпису з відповідними контрзаходами. Атака, яка буде розглядатися в даному дослідженні – BEARZ – є модифікацією атаки на геш-підписову схему DLP розроблену Espitau (Espitau et al., 2016) (Espitau et al., 2018).

Хоча відбірник алгоритму Falcon є вразливим до атак помилками, і це дає змогу отримати приватний ключ, ефективність компонентів даного алгоритму сприяє тому, що він є конкурентноспроможним, навіть при застосуванні контрзаходів.

В середньому, у випадку схеми Falcon, контрзаходи викликають лише до 5% зниження ефективності. Одним з ефективних заходів проти пропонованої атаки помилками є нульова перевірка. Для набору параметрів з високою захищеністю подібний контрзахід спричиняє зниження продуктивності до 16%.

**2 Схема підпису Falcon**

Falcon – геш-підписова схема, яка заснована на шифруванні на основі DLP та методах швидкої вибірки Фур’є. Завдяки цим методам, вдається забезпечити невеликі розміри закритого ключа та підвищену швидкодію процедури вибірки. Генерація ключів відбувається за Алгоритмом 1 (Рис.2.1). Після генерації ключів, отримуємо NTRU-поліноми. При цьому - особисті ключі, а – відкритий ключ.

Процес підписання показано в Алгоритмі 2 (Рис.2.1). Він відбувається шляхом знаходження короткого вектору в решітці NTRU з використанням закритого ключа. Для цього застосовується процедура ffSampling – показана в Алгоритмі 3 (Рис.2.2).

Перевірка підпису виконується шляхом перевірки модуля, який розміщується в межах необхідної межі β, за допомогою відкритого ключа. Ця процедура показана в Алгоритмі 4 (Рис.2.2).

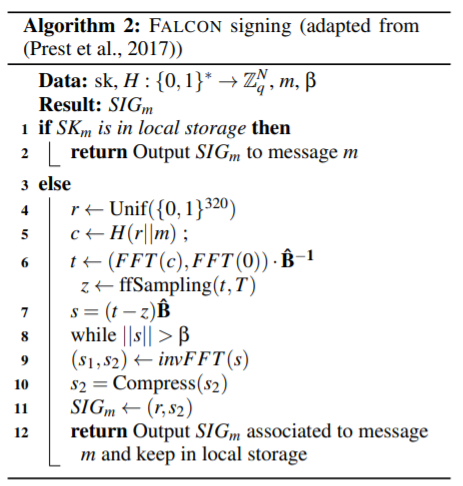
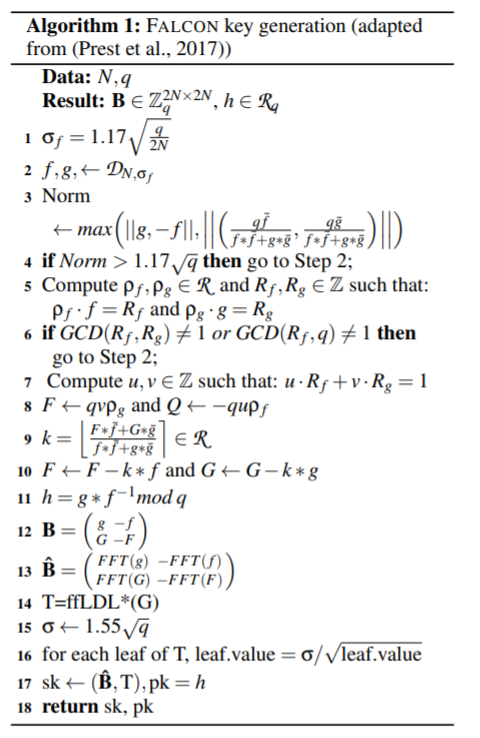


Рис. 2.1 - Алгоритми генерування підпису і підписування (Алгоритм 1 та 2)

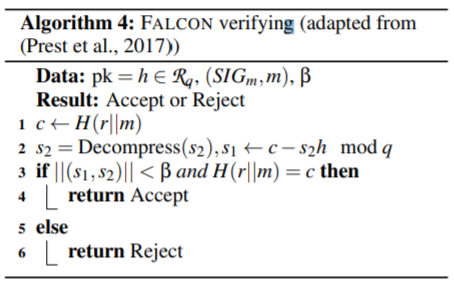
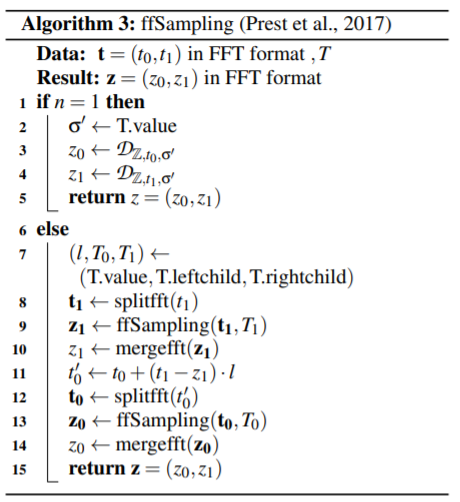


Рис. 2.2 - Алгоритм ffSampling та алгоритм перевірки (Алгоритм 3 та 4)

**3 Часові атаки на Falcon**

Деякі математичні компоненти, які використовуються в алгоритмі ЕЦП Falcon, є вразливими до атак, заснованих на аналізі часових показників. Це пов’язано з певними складнощами реалізації цих компонентів за постійний час.

**3.1 Відбірник Гауса**

Саме Гаусові відбірники, як правило, є вразливим місцем для часових атак проти криптографії на решітках. Для схеми Falcon Гаусів відбірник виконує роль відбірника коротких векторів у решітках.

**3.2 Теоретичне перетворення числа (NTT)**

NTT є потенційно вразливим до часових атак, оскільки використовує велику кількість модульної арифметики, яку важко реалізувати за постійний час. У схемі Falcon NTT використовується для пришвидшення множення кільцевих многочленів.

**3.3 HashtoPoint**

Процес гешування повідомлення до точки в кільці поліномів „може бути складно здійсненним при постійному часі” – зазначають автори алгоритму Falcon. Тому функція є потенційно вразливою до часових атак.

**3.4 Запропоновані контрзаходи проти часових атак**

Одним з ефективних на даний момент заходів проти часових атак є алгоритм BlindVector, запроваджений Saarinen (2017). У BlindVector покращено випадкові перемішування для протистояння атакам по бічних каналах. Цей алгоритм використовується у даній реалізації, крім того гарантується постійна за часом реалізацію навіть для подальшого збільшення складності атак.

Для протидії часовим атакам, функції NTT та FFT також реалізовані постійними за часом. Це стає можливим завдяки обробці всіх необхідних змінних гілки та відмови від логіки для більш тривалих арифметичних операцій. Для зменшення впливу на продуктивність даних контрзаходів, використовується «ледаче» скорочення (lazy reduction) та векторизація SIMD. Для забезпечення можливості виконання множення двох кілець у домені NTT застосовується поточкове модульне множення, яке забезпечує обчислення кожного елементу незалежно. Дані операції є послідовними та безумовними, тому можна говорити про їх усталеність у часі. Ще однією особливістю є те, що у перетворенні застосовується вдосконалений NTT алгоритм Cooley-Tukey. Покращення застосовуються для підвищення продуктивності, увімкнення автоматичної векторизації та гарантування того, що не використовуються умовні операції розгалуження.

Також серед запропонованих заходів є відкидання зразку. Даний конрзахід полягає в наступному – з випадкових адрес зчитується додатковий кеш з метою спотворити статистику для SCA і після цього зайві зчитування відкидаються. Діапазон норм відкидань у реалізації, що розглядається, обрано на рівні 6,25%, 12,5% та 25%.

Ще одним нововведенням є ефективна конструкція методики гаусової вибірки (CDT) з постійним часом. Перевагою даної конструкції є чітка кількість операцій пошуку-зчитування, а також використання однакової арифметики, незалежно від гілки, обраної згідно з CDT. Верхня межа кількості необхідних пошукових операцій дорівнюватиме тому кожен виклик пробовідбірника доповнюється до найближчого ступеня двійки для того, щоб виконати таку ж кількість тактових циклів.

**4 BEARZ атака на Falcon**

BEARZ – атака помилками на схему ЕЦП Falcon, запропонована Sarah McCarthy та ін. (2019). Принцип дії атаки полягає у вилученні базису через переривання рекурсії або обнулення. Атака добуває закриту ключову інформацію, шляхом спричинення помилок в роботі. Модель зловмисника передбачає, що він може пропускати команди та обнуляти змінні.

**4.1 Рекурсія алгоритму Falcon**

Схема Falcon має два рекурсивні виклики на верхньому рівні алгоритму ffSampling. Для даного цільового вектора , алгоритм діє спочатку на , рекурсивно, справа наліво, а потім на . Кожен елемент - і потім - безперервно ділиться на два вектори довжини поки не буде дорівнювати 2, а потім відбирає коефіцієнти Гаусового розподілу, щоб отримати вектор вибірки . Через це алгоритм верхнього рівня містить дві рекурсивні гілки. Зауважимо, що всі процедури виконуються в домені FFT. Для вектора довжини , перші значень будуть представлені реальними коефіцієнтами, а другі - уявними.

Спираючись на структуру рекурсії Falcon, для успішної атаки слід було б перервати рекурсивний виклик у необхідній точці, щоб тільки місць були б заповнені. На жаль це не є можливим через характер функцій FFT.

**4.2 FFT: злиття та розділення**

Функції злиття та розділення для схеми Falcon виконуються у домені FFT. Це спричиняє проблему для атаки помилками, оскільки нульові вхідні дані не представлені нулем у FFT. Тому є необхідність відстеження коефіцієнтів зразків під час руху вгору по рекурсивному дереву. Після отримання вектору решітки із Гаусового зразка, до нього застосовується зворотня функція FFT (. Це необхідно, щоб переконатися, що підпис не знаходиться в домені FFT. Після цього може бути використана та сама після-обробка що і для DLP-атаки.

Атаки шляхом переривання рекурсії можна класифікувати, в залежності від місця переривання. Але усі методи атаки призводять до однакового вихідного векторного формату для ; перші коефіцієнів примусово задаються нулями.

**4.2.1 Переривання другої рекурсії (для )**

Атака полягає в перериванні виклику алгоритму вибірки в кінці, після першого рекурсивного виклику (рядок 10 алгоритму 3). Після виконання атаки заповниться вибраними коефіцієнтами, а залишиться повністю нульовим. Це також можна виконати для , з метою обнулити перші коефіцієнтів.

**4.2.2 Обнулення або атака пропуску (для )**

Можливі два варіанти даної атаки. Перший - при передостанньому злитті ( = Від 256 до 512), встановіть необхідні (наприклад, перша половина, якщо ) з вихідних коефіцієнтів в нуль шляхом пропуску операцій або обнулення. Операції, що пропускаються: , з коду (Prest et al., 2017)( merge\_fft() функція у falcon\_fft.c). Другий - встановити необхідну кількість перших коефіцієнтів в нуль перед обчисленням відповідного вектора решітки, тобто перезаписати вихід відбірника - .

**4.2.3 Переривання середньої рекурсії ( )**

Даний тип атаки вимагає одноразового попереднього обчислення, проте дозволяє застосовувати помилку на стадії одновимірного Гаусового відбірника, що сприяє простоті фізичного вбудовування.

Якщо нам необхідно прирівняти до нуля ліву половину , то вектор лівої сторони (LHS) при останньому виклику merge\_fft() повинен містити у першій половині нульові значення, а у векторі правої сторони (RHS) перша половина коефіцієнтів має бути рівною другій половині. Кожен реальний коефіцієнт вектору генерується як та , де - розмірність вектора вищого рівня, і . Для того, щоб прирівняти і до нуля, встановлюється і для кожного . Тому, наприклад, для = 512, перші = 128 коефіцієнтів 256-мірного вектора LHS встановлюються рівними нулю, а перші 128 коефіцієнтів 256-мірного вектора RHS встановлюються рівними другим 128 коефіцієнтам цього ж вектора.

Щодо 256-розмірного RHS вектору - він повинен містити в першій половині значення, рівні значенням другої половини. Щоб це могло статися, 128-мірний вектор LHS повинен мати реальні значення, що дорівнювали б уявним значенням, а 128-розмірний RHS вектор повинен бути нульовим. Це зумовлено тим, що необхідно отримати та рівним для кожної ітерації . Функція merge\_fft () виконує це наступним чином: і . Щоб умова вище виконувалася, встановлено наступні правила - перша половина коефіцієнтів дорівнюватиме другій половині, і . Тому рівняння будуть просто залежати від вектора подачі LHS.

Для RHS кожну нижню гілку можна встановити в нуль. Візьмемо LHS 128-розмірний вектор: 64-мірний вектор LHS повинен мати рівні реальні та уявні значення, а RHS має бути нульовим. Будь-який рівний нулю вектор повинен мати обидва вектори подачі рівними нулю, тому гілки нижче цієї можуть бути обнулені. Цей метод може застосовуватися для будь-якого такого, що , де .

**4.3 Обробка після нападу**

Після отримання недійсного підпису атака передбачає відновлення таємного базису з нього. Наприклад, у нас мається коефіцієнтів вектора , які дорівнюють нулю. Тоді обчислюємо підпис Falcon як:

де - базова матрицею в домені FFT. Але нам потрібна лише друга половина - , де:

і так як встановлено в нуль:

Крім того, перші коефіцієнтів дорівнюють нулю, що означає:

і оскільки деякі з коефіцієнтів дорівнюють нулю, ми нарешті маємо:

Тим самим отримується підрешітка решітки, згенерованої за допомогою F. Таким чином, з кількома недійсними підписами можна знайти решітку, породжену , (де є коротким вектором у цій решітці), а завдяки алгоритму BKZ можна знайти цей короткий вектор. Після цього можна отримати з відкритого ключа , і таким чином можна знайти таємний базис NTRU решітки.

**4.4 Результати атаки**

Після успішного виконання всіх тестів, було зібрано недійсних підписів для значень нулів, наведених в таблиці 2. Для отримання полінома приватного ключа було застосовано алгоритм BKZ (FPLLL Development Team, 2016). Таблиця 1 містить часові результати запуску алгоритмів.

Таблиця 1 – Часи (у секундах) на застосування алгоритму BKZ для недійсних підписів для отримання базового полінома (працює на одному ядрі процесора Intel Core i7-6700HQ на частоті 2,60 ГГц.)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Набір параметрів | m | l=m+1 | l=m+2 | l=m+3 |
| Набір 1 | 64 | 0.07 | 0.07 | 0.08 |
| 128 | 0.35 | 0.30 | 0.36 |
| 256 | 2.17 | 2.80 | 2.36 |
| Набір 2 | 128 | 0.28 | 0.29 | 0.35 |
| 256 | 2.24 | 2.55 | 2.67 |

**4.5 Модель помилки**

Модель помилки передбачає аналіз бічних каналів, завдяки якому може бути виявлено вікно між першим і другим рекурсивним викликом. В межах цього вікна алгоритм може бути перервано.

Для застосування атаки обнуленням можна використати момент зберігання вектору в оперативній пам’яті, обнуливши під час цього необхідні біти. (Naccache et al., 2005). Як альтернативу можна розглядати пропуск рядків. Його можна виконати за допомогою перепадів тактової частоти процесора. (Blömer et al.,2014).

**4.6 Контрзаходи проти BEARZ**

Для запобігання атаці BEARZ існують певні контрзаходи. Одним з найпростіших методів виявлення атак помилками є подвійне обчислення підпису. Це подвоює час підписання. При перевірці відразу після підписування підписувач може переконатися в тому, що обладнання не піддавалося атакам помилками. (Bruinderink та Pessl, 2018). Також простим та ефективним методом виявлення ВEARZ атаки є перевірка того, що вибраний вектор не йде до нуля в певній точці вздовж своєї довжини в кінці ffSampler алгоритму.

**5 Результати та оцінка**

**5.1 Falcon з контрзаходами**

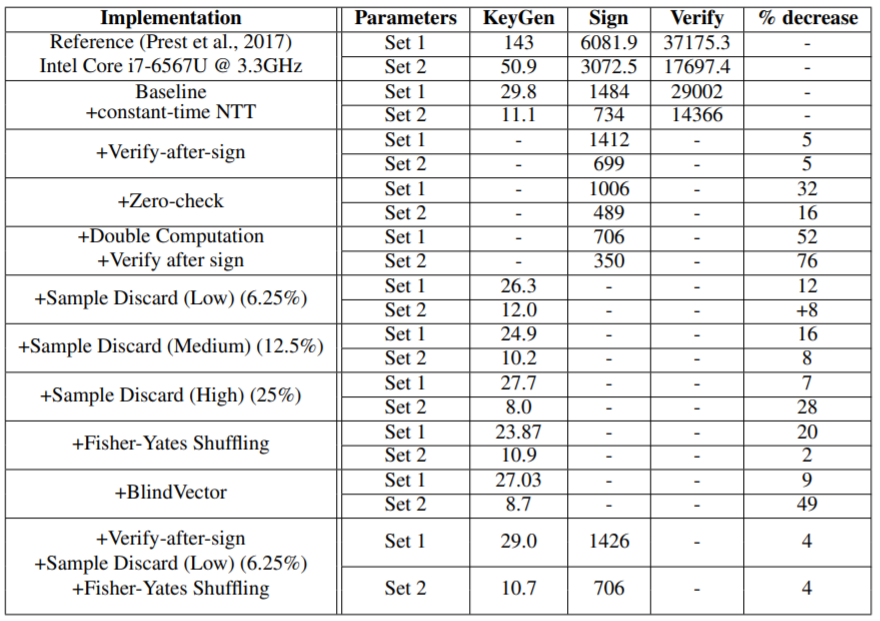


Рис. 5.1 - Результати продуктивності (ops / sec) для Falcon із контрзаходами SCA, включаючи відсоткове зменшення задіяних компонентів, на Intel E5-1620 при 3,7 ГГц.

Як видно з рисунку 5.1 – контрзаходи не мають значного впливу на швидкодію Falcon. Загальна рекомендована комбінація контрзаходів уповільнює генерацію ключів та підписання менш ніж на 5% для набору параметрів 1. Такий результат зумовлений ефективністю процесів вибірки та перевірки.

**5.2 Ефективність контрзаходів**

**5.2.1 Часові атаки**

Для перевірки контрзаходу проти атак помилками, було 100 разів проведено процедуру підписання та визначено, що для набору 1, він потрапляв в діапазон 50 операцій на секунду, а для набору 2 - 20 операцій на секунду. Це свідчить про його ефективність проти часових атак.

**5.2.2 BEARZ Атака**

Перевірка після підписування може не виявити атаки помилками, яка націлена на вибірку, якщо одна і та ж помилка успішно реалізована двічі. Тому атака все ще може сформувати дійсний підпис і залишитись невиявленою. Однак контрзахід нульової перевірки повинен виявити атаку зі 100% успіхом. Тому його можна рекомендувати як мінімальний і достатній контрзахід.

**5.3 Порівняння з іншими схемами підписів на основі решітки**

У цьому підрозділі проводиться порівняння результатів даної реалізації Falcon з Dilithium (Lyubashevsky et al., 2017) та Bliss-B (Ducas, 2014) з використанням пропонованих контрзаходів. Застосування перевірки після підпису спричиняє менші втрати продуктивності для алгоритму Falcon порівняно з алгоритмом Dilithium, як показано на рисунку 5.3. Навіть при високому рівні безпеки його продуктивність знижується не так помітно, як у BLISS-B, як показано на рисунку 5.2. Це можна пояснити ефективністю компонентів алгоритму ЕЦП Falcon.

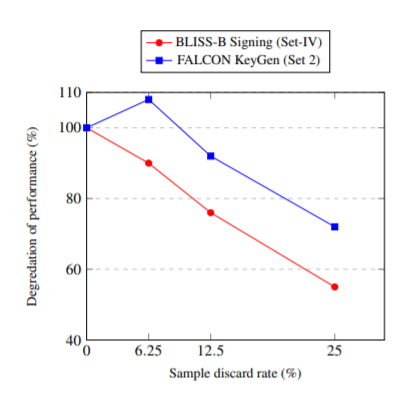


Рис. 5.2 – Ефект контрзаходу відкидання зразків на BLISS-B та Falcon для відповідного (192 біти) рівня захищеності

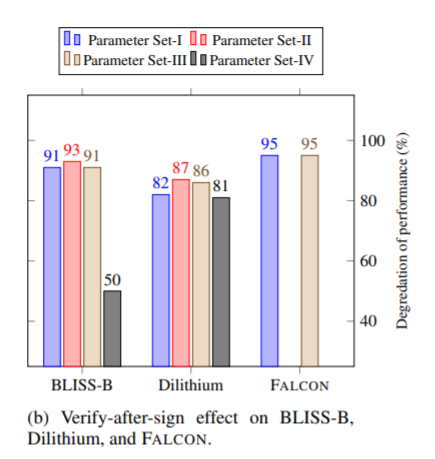


Рис. 5.3 – Ефект перевірки після підпису на BLISS-B, Dilithium та Falcon

BLISS-B пропонує широкий спектр контрзаходів для захисту від запропонованих атак, і, як видно з рисунку 5.3, їх вплив на продуктивність може коливатися від додаткових 10% до 50%, залежно від реалізації. Однією з переваг схеми Dilithium є захищеність від атак на відбірник Гауса, проте його захист від атак помилками Bruinderink та Pessl (2018) залишається проблемою. При застосуванні до нього контрзаходів, його ефективність сповільнюється майже на 20%. А це є навіть гіршими показниками, ніж показники BLISS-B при застосуванні контрзаходу для захисту Гаусового відбірника.

**6 Висновки**

У цьому дослідженні було розглянуто атаку на схему підпису Falcon - BEARZ. Було показано, що певні математичні компоненти, які використовуються в алгоритмі ЕЦП Falcon, є вразливими до атак, заснованих на аналізі часових показників, а також, що Falcon є вразливим до атак помилками на відбірник Гауса. Через це при стандартизації чи впровадженні слід розглядати можливість фізичних атак. Було розглянуто можливі контрзаходи для протидії спеціальним атакам, показано вплив даних контрзаходів на швидкодію алгоритму. Крім того, було порівняно швидкодію різних схем, заснованих на решітках, і показано, що Falcon є конкурентним кандидатом другого раунду, навіть із застосуванням запропонованих контрзаходів.

**Список джерел**

[1] Ajtai, M. and Dwork, C. (1997). A public-key cryptosystem with worst-case/average-case equivalence. In STOC ’97 Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pages 284–293.

[2] Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., and Smith-Tone, D. (2019). Status Report on the First Round of the NIST PostQuantum Cryptography Standardization Process. http://aiweb.techfak.uni-bielefeld.de/ content/bworld-robot-control-software/. [Online; accessed February 2019].

[3] Alkim, E., Ducas, L., Pöppelmann, T., and Schwabe, P. (2016). Post-quantum key exchange-a new hope. In USENIX Security Symposium, volume 2016. Bindel, N., Buchmann, J., and Krämer, J. (2016). Lattice-based signature schemes and their sensitivity to fault attacks. In Fault Diagnosis and Tolerance in Cryptography (FDTC), 2016 Workshop on, pages 63–77. IEEE.

[4] Bindel, N., Kramer, J., and Schreiber, J. (2017). Special session: hampering fault attacks against lattice-based signature schemes-countermeasures and their efficiency. In Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2017 International Conference on, pages 1–3. IEEE.

[5] Blömer, J., Silva, R. G. D., Günther, P., Krämer, J., and Seifert, J.-P. (2014). A practical second-order fault attack against a real-world pairing implementation. In Fault Diagnosis and Tolerance in Cryptography (FDTC), 2014 Workshop on, pages 123– 136. IEEE.

[6] Bruinderink, L. G., Hülsing, A., Lange, T., and Yarom, Y. (2016). Flush, Gauss, and reload-a cache attack on the BLISS lattice-based signature scheme. In International Conference on Cryptographic Hardware and Embedded Systems, pages 323–345. Springer. Bruinderink, L. G. and Pessl, P. (2018). Differential fault attacks on deterministic lattice signatures. IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 21–43.

[7] Ducas, L. (2014). Accelerating BLISS: the geometry of ternary polynomials. Cryptology ePrint Archive, Report 2014/874. https://eprint.iacr.org/ 2014/874.

[8] Ducas, L., Lyubashevsky, V., and Prest, T. (2014). Efficient identity-based encryption over ntru lattices. In International Conference on the Theory and Application of Cryptology and Information Security, pages 22–41. Springer.

[9] Ducas, L. and Prest, T. (2016). Fast fourier orthogonalization. In Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, pages 191–198. ACM.

[10] Espitau, T., Fouque, P., G’erard, B., and Tibouchi, M. (2018). Loop-abort faults on lattice-based signature schemes and key exchange protocols. IEEE Transactions on Computers, 67(11):1535–1549.

[11] Espitau, T., Fouque, P.-A., Gérard, B., and Tibouchi, M. (2016). Loop-abort faults on lattice-based fiatshamir and hash-and-sign signatures. In International Conference on Selected Areas in Cryptography, pages 140–158. Springer.

[12] Fisher, R. A., Yates, F., et al. (1938). Statistical tables for biological, agricultural and medical research. Statistical tables for biological, agricultural and medical research.

[13] FPLLL Development Team, O. T. (2016). fplll, a lattice reduction library. Available at https:// github.com/fplll/fplll.

[14] Gentry, C. and Boneh, D. (2009). A fully homomorphic encryption scheme, volume 20. Stanford University Stanford.

[15] Hodgers, P., Regazzoni, F., Gilmore, R., Moore, C., and Oder, T. (2016). State-of-the-art in physical side-channel attacks and resistant technologies. Technical report.

[16] Howe, J., Khalid, A., Rafferty, C., Regazzoni, F., and O’Neill, M. (2016). On practical discrete Gaussian samplers for lattice-based cryptography. IEEE Transactions on Computers.

[17] Howe, J., Pöppelmann, T., O’Neill, M., O’Sullivan, E., and Güneysu, T. (2015). Practical lattice-based digital signature schemes. ACM Transactions on Embedded Computing Systems (TECS), 14(3):41.

[18] Karmakar, A., Roy, S. S., Reparaz, O., Vercauteren, F., and Verbauwhede, I. (2018). Constant-time discrete gaussian sampling. IEEE Transactions on Computers.