УДК 621.391:519.2 **DOI:10.30837/rt.2024.4.219.04**

Є.В. КОТУХ, канд. техн. наук, Г.З. ХАЛІМОВ, д-р техн. наук, І.Є. ДЖУРА, Г.О. ХІВРЕНКО

ЗАСТОСУВАННЯ СХЕМИ ШИФРУВАННЯ LINE В МЕХАНИЗМІ ІНКАПСУЛЯЦІЇ КЛЮЧІВ ДЛЯ ПРОТОКОЛУ АВТЕНТИФІКАЦІЇ В МЕРЕЖАХ 5G

Вступ

Мережа 5G є ключовим двигуном цифрової трансформації та Четвертої промислової революції. Послуги, які пропонує платформа 5G, є синергетичними та масштабованими, що дозволяє значно збільшити швидкість передачі даних за допомогою різних технологій радіодоступу (RAT). Мережа 5G надає високоякісні послуги, включаючи значне збільшення кількості запитів на підключення, високу швидкість передачі та обробки даних до 20 Гбіт/с, а також зменшену затримку мережі до 1 мс, навіть за умов мобільності користувача.

Технологія 5G дозволяє компаніям підключати більше пристроїв із швидшим обміном інформацією, що призводить до підвищення потенційної вразливості та значного розширення векторів загроз і атак. Питання безпеки та конфіденційності, такі як підробка мережі та відсутність конфіденційності в попередніх поколіннях RAN, були ретельно вивчені експертами з безпеки. Щоб вирішити ці проблеми, органи стандартизації 3GPP визначили протокол і процедури автентифікації AKA та керування ключами. Вони включають взаємну автентифікацію між пристроями користувача та мережею, цілісність і конфіденційність сигналів, а також отримання криптографічних ключів для захисту даних площини U та C.

3GPP визначив такі три методи автентифікації для мереж 5G:

- 5G-AKA: автентифікація 5G і керування ключами;
- EAP-AKA: розширюваний протокол автентифікації – автентифікація та керування ключами;
- EAP-TLS: розширюваний протокол автентифікації – безпека транспортного рівня.

Однак існують значні вразливості в API-інтерфейсах мережі 5G, зокрема щодо слабкого захисту особистих даних користувачів і можливості несанкціонованого доступу до пристроїв IoT. Ці уразливості є критичними для систем дистанційного керування. В протоколі 5G-AKA були виявлені недоліки і, зазвичай, вважається, що AKA не забезпечує належного захисту персональних даних від активних зловмисників [1 – 3]. Відзначено, що AKA необхідно наділити необхідними властивостями безпеки, які наразі відсутні. Було запропоновано кілька рішень на рівні протоколу автентифікації, що включають інкапсуляцію ключів (наприклад, EAP-AKA, 5G-AKA-FS, SUCI-AKA, 5G-IPAKA, AKA-LCCO) [2 – 7].

Поточний стандарт 5G не вирішує проблем, пов’язаних з квантовими обчисленнями, і продовжує покладатися на традиційні криптографічні методи, такі як криптографія на основі еліптичної кривої (ECC). Однак проблему дискретного логарифмування еліптичної кривої (ECDLP) можна розв’язати за поліноміальний час за допомогою квантових комп’ютерів, що становить значну загрозу безпеці.

Одним із можливих рішень для усунення цих уразливостей є впровадження квантово-захищеного шифрування для розробки протоколів автентифікації та координації ключів у мережах 5G. Враховуючи вимоги 3GPP щодо безпеки на рівні 128-бітного AES, автори [8] проаналізували ефективність протоколу автентифікації 5G-AKA з використанням механізму постквантової інкапсуляції ключів (KEM) з четвертого раунду конкурсу NIST. Однак схеми шифрування на основі коду, такі як McEliece і NTRU на основі решітки, демонструють низьку ефективність і великі розміри ключів порівняно зі схемами на основі проблеми дискретного логарифмування еліптичної кривої (ECDLP). Заміна поточної асиметричної криптографії постквантовими безпечними схемами призведе до витрат як з точки зору зв’язку, так і ефективності роботи мережі. Потрібні подальші дослідження, щоб визначити відповідне застосування постквантової захищеної криптографії для задоволення очікуваних показників продуктивності та функціональності архітектур 5G і 6G наступного покоління. Досягнення цієї мети вимагає дослідження нових, більш ефективних постквантових асиметричних схем.

Одне з можливих рішень для масштабованої секретності на основі криптосистеми з логарифмічними підписами та багатопараметричними групами наведено в [16 – 19]. Показано, що на групах великого порядку та малої розмірності можна вирішити задачу оптимізації накладних витрат за рахунок менших ключів.

Розвиток квантових обчислень прогресує як у практичних застосуваннях, так і у таких як тестування квантових комп'ютерів такими компаніями, як IBM і Google, так і в розробці алгоритмів, призначених для квантових комп'ютерів, таких як алгоритм Шора [14] і алгоритм Гровера [15]. Традиційні обчислювальні проблеми, які лежать в основі безпеки алгоритмів обміну ключами, такі як проблема цілочисельної факторизації, можуть бути вирішені за поліноміальний час зловмисниками, оснащеними квантовими комп'ютерами.

Національний інститут стандартів і технологій (NIST), який традиційно впливав на стандартизацію та впровадження криптографічних шифрів, таких як DES і AES, зараз бере участь у довгостроковому процесі стандартизації квантово-стійких шифрів [10]. Серед фіналістів четвертого раунду конкурсу KEM NIST, обраних для подальшої стандартизації, увійшли CRYSTALS-Kyber, BIKE, Classic McEliece і HQC. Стаття [8] надає оцінки обчислювальних витрат і розмірів параметрів для постквантових схем KEM. Відзначається, що Classic McEliece несе найвищі обчислювальні витрати. Однак Classic McEliece швидший за ECIES Secp256r1, тоді як Kyber перевершує обидва профілі ECIES за швидкістю. ECIES пропонує невелику вартість зв'язку порівняно з постквантовими KEM, тоді як Classic McEliece має найвищу вартість зв'язку через великий розмір відкритого ключа.

Актуальним завданням є вибір квантово-стійкої схеми для заміни ECIES найближчим часом. У статті запропоновано алгоритм інкапсуляції ключів на основі криптосистеми LINE з логарифмічними підписами для розробки протоколів автентифікації та координації ключів у мережах 5G. Використання спрямованого шифрування LINE пропонує переваги в масштабованості та неоднорідності, оптимізуючи обчислювальні і операційні витрати мережі.

Інтегрована схема шифрування на основі еліптичних кривих

Схема шифрування інтегрованої еліптичної кривої (ECIES) [9] є добре відомою гібридною схемою шифрування, що включає механізм інкапсуляції ключів (KEM) і механізм інкапсуляції даних (DEM), що дозволяє шифрувати повідомлення довільної довжини. Ця схема є ключовим компонентом протоколу 5G-AKA. ECIES – KEM має наступні три алгоритми:

$KeyGen(p)$: під час введення загальнодоступного параметра p алгоритм виводить пару відкритого та закритого ключів (pk, sk) так, що $pk = sk \cdot G$, де p – параметр еліптичної кривої, стандартизований у secp256r1, а $G \in p$ – базова точка;

$Encap(pk)$: під час введення відкритого ключа pk алгоритм генерує пару відкритих і закритих ключів сеансу (R, r), а $R = r \cdot G$ потім виводить зашифрований текст $C_0 = R$ і спільний ключ $ks = KDF(r \cdot pk)$, де KDF функція виведення ключа.

$Decap(sk, C_0)$: при введенні зашифрованого тексту C_0 та закритого ключа sk алгоритм видає загальний ключ $ks = KDF(sk \cdot C_0)$.

Для шифрування в 5G AKA TS 33.501 дозволяє використовувати два профілі ECIES, а саме Curve25519 і Secp256r1.

Механізм інкапсуляції ключів для постквантових алгоритмів залишається тим самим, але з іншими алгоритмами $KeyGen()$, $Encap()$, $Decap()$.

Реалізація спрямованого шифрування в механізмі інкапсуляції ключів

Ми пропонуємо реалізацію механізму інкапсуляції ключів (KEM) на основі спрямованого шифрування в криптосистемі LINE [11]. Побудова цієї криптосистеми базується на відомій алгебраїчній задачі, яка стверджує, що єдине рішення існує лише для повністю визначеної системи лінійних рівнянь. Коли система рівнянь визначена неповністю, кількість розв'язків визначається множиною можливих розв'язків. Єдина можлива атака на криптосистему передбачає вичерпний пошук і визначення змінних у системі лінійних рівнянь.

Основним криптографічним примітивом в алгоритмі є логарифмічний підпис, який реалізує безключове шифрування з властивістю факторизації ключа. Система лінійних рівнянь пов'язує значення логарифмічних сигнатур, які служать її змінними, а безпека криптосистеми визначається міцністю набору рішень [12]. Спрямоване шифрування в криптосистемі LINE складається з наступних етапів:

- налаштування загальних параметрів криптосистеми;
- побудова секретного логарифмічного підпису над $F(2^m)$ полем;
- створення відкритих і закритих ключів криптосистеми;
- побудова неповної системи лінійних рівнянь для логарифмічних сигнатур;
- шифрування.

Реалізація алгоритму KeyGen

Ми повинні встановити рівень секретності та побудувати загальні параметри криптосистеми. Ми повинні виконати наступні п'ять кроків.

Крок 1. Налаштування загальних параметрів криптосистеми

Параметри криптосистеми LINE наступні:

– розмір слів, поданих над полем $F(2^m)$, m = 8,16,.. ,64;

– значення L, K, де L – кількість рівнянь, а K – кількість параметрів системи лінійних рівнянь.

Значення mL визначає довжину узгодженого секретного ключа ks в алгоритмі KEM на основі криптосистеми LINE, значення m (K – L) – секретність криптосистеми LINE щодо рейдерських атак за рахунок визначення система лінійних рівнянь.

Розглянемо приклад 64-розрядної криптографії для постквантового KEM. Зафіксуємо обчислення для 8-розрядних слів (m =8) і, відповідно, отримаємо значення L =64/8=8 і K=16. Ми визначимо криптосистему LINE на основі 16 8-розрядних логарифмічних сигнатур, які з’єднані в систему з 8 лінійних рівнянь.

Крок 2. Побудова секретного логарифмічного підпису

Секретна факторизована логарифмічна сигнатура будується на основі простої логарифмічної сигнатури β_1 типу $(\overbrace{2,\ldots,2}^{m})$ за допомогою наступних секретних гомоморфних перетворень [13] $\beta_1 \xrightarrow{\rho_1} \beta_2 \xrightarrow{\rho_2} \beta_3 \xrightarrow{\rho_3} \beta_4 \xrightarrow{\rho_4} \beta_5 \xrightarrow{\rho_5} \beta$, де

ρ_1 – змішування записів у блоках підписів β_2;

ρ_2 – таємне перетасування блоків підписів β_3;

ρ_3 – рандомізація записів масиву підписів на основі секретної матриці $\beta_4(i) = \varpi_{m\times m} \oplus \beta_3(i)$;

ρ_4 – секретне гомоморфне перетворення рядків масиву $\beta_5(i) = \gamma \cdot \beta_4(i)$, $i = \overline{1,2m}$, $\gamma \in F(2^m)$;

ρ_5 – секретне гомоморфне перетворення рядків масиву $\beta(i) = \beta_5(i) \cdot \omega_{m\times m}$, $i = \overline{1,2m}$ $\omega_{m\times m}$ секретна оборотна двійкова матриця розмірності $m \times m$.

Нехай ρ_1 =: [11011110], ρ_2 =: [62054371],

$\gamma = 1 + x + x^3 + x^5 + x^7$

$$\varpi_{m\times m} = \begin{vmatrix} 00111110 \\ 00101001 \\ 01110001 \\ 01010000 \\ 10001010 \\ 01010010 \\ 11101111 \\ 10000011 \end{vmatrix} \qquad \omega_{m\times m} = \begin{vmatrix} 10000010 \\ 00011000 \\ 10100110 \\ 10110000 \\ 00111011 \\ 01111110 \\ 10011000 \\ 00010101 \end{vmatrix}$$

Результати обчисленої логарифмічної сигнатури для заданого дисплея представлені в табл. 1. Початкова проста логарифмічна сигнатура складається з 8 блоків по два записи в кожному, що відповідає типу $(\overbrace{2,\ldots,2}^{m})$. Блоки в таблиці розділені підкресленням.

Таблиця 1

Представлення логарифмічної сигнатури

β_1	β_2	β_3	β_4	β_5	$ls = \beta$
00000000	10000000	00000000	00111110	00101001	10001000
10000000	00000000	00100000	00011110	01000000	00011000
00000000	01000000	01001110	01100111	11011101	01111010
01000000	00000000	01000111	01101110	01110111	11111101
00000000	00000000	01000000	00110001	00001001	00101110
00100000	00100000	00000000	01110001	11011011	10011100
00000000	01110000	00111100	01101100	11011110	11110111
01110000	00000000	11011000	10001000	10010011	10111111
01000000	10111000	10111000	00110010	01001100	01011101
10111000	01000000	01000000	11001010	11101000	00000111
11011000	00111100	01110000	00100010	11000000	10011010
00111100	11011000	00000000	01010010	11110111	01111111
10110100	01111110	10000000	01101111	10011011	10000100
01111110	10110100	00000000	11101111	01001110	11000101
01001110	01001110	01111110	11111101	01101011	00001000
01000111	01000111	10110100	00110111	10000011	00001111

Для криптосистеми з параметрами L = 8 і K = 16 необхідно побудувати K = 16 масивів логарифмічних сигнатур. ls_k, $k = \overline{1,K}$: L=8 факторизований і KL=8 нефакторизований.

Крок 3. Побудова секретних ключів криптосистеми

Секретними ключами криптосистеми LINE є масиви $(\alpha_{i,j})_k$, $(t_j)_k$, $(\tau_j)_k$, $i = \overline{1,2}$, $j = \overline{1,m}$, $k = \overline{1,K}$ і таємне перетворення $\psi_{m\times m}$. Індекс $i = \overline{1,2}$ показує кількість записів в одному блоці логарифмічного масиву, індекс $j = \overline{1,m}$ – кількість блоків у масиві ls. Записи секретних масивів розпізнаються за m бітовими словами. Для нашого прикладу в табл. 2 представлено згенерований $(\alpha_{i,j})$, (t_j), (τ_j) для першого логарифмічного підпису ls_1.

Таблиця 2

Параметри логарифмічної сигнатури

$(\alpha_{i,j})$	(t_j)	(τ_j)	$\psi_{m\times m}$
10100000	00100101	00001100	00010100
00010011	10000100	00101001	00001011
11011011	11110110	10010101	01001110
11111101	01011000	11110111	01001001
11101001	01110111	11001111	10010001
11011111	11011000	10011010	00000110
11110010	10110101	01111111	10100001
00001000	11111001	11101110	00000011
10000100			
01111000			
10000110			
00010010			
11111101			
00101110			
11100011			
01111111			

Для криптосистеми з K=16 логарифмічних масивів необхідно побудувати 16 масивів $(\alpha_{i,j})_k$, $(t_j)_k$, $(\tau_j)_k$.

Крок 4. Побудова відкритих ключів криптосистеми

Відкритими ключами криптосистеми LINE є масиви $(\gamma_{i,j})_k$ і $(\lambda_{i,j})_k$ $i = \overline{1,2}$, $j = \overline{1,m}$, $k = \overline{1,K}$. Побудова масивів визначається виразами для логарифмічних індексів, що розкладаються на фактори ls_k, $k = \overline{1,L}$:

$$(\gamma_{i,j})_k = (\beta_{i,j})_k + (t_j) + (\alpha_{i,j})_k \psi, \ (\lambda_{i,j})_k = (\alpha_{i,j})_k + (\tau_j)_k \quad (1)$$

і для нефакторизованих логарифмічних підписів ls_k, $k=\overline{L+1,K}$

$$(\gamma_{i,j})_k=(\beta_{i,j})_k\psi+(t_j)_k,\ (\lambda_{i,j})_k=(\beta_{i,j})_k+(\tau_j)_k \tag{2}$$

Табл. 3 показує розраховані $(\gamma_{i,j})$, $(\lambda_{i,j})$ згідно з виразами (1), (2) для першої логарифмічної сигнатури ls_1. Для криптосистеми з K=16 логарифмічних масивів необхідно побудувати 16 масивів $(\gamma_{i,j})$ і $(\lambda_{i,j})$.

Таблиця 3

Побудова масивів відкритих ключів

$(\gamma_{i,j})$ для ls_1	$(\lambda_{i,j})$ для ls_1
11110111	10101100
11010110	00011111
10011011	11110010
11110101	11010100
00011011	01111100
00001001	01001010
00010110	00000101
01110110	11111111
00111000	01001011
11101101	10110111
11110001	00011100
01001111	10001000
10111101	10000010
00001000	01010001
00000010	00001101
11001111	10010001

Крок 5. Побудова неповної системи лінійних рівнянь для логарифмічних сигнатур

Ми реалізували випадковий алгоритм побудови системи з L лінійних рівнянь і K параметрів. Ми згенерували двійкову випадкову матрицю $K\times K$ рангу K. Будь-які L членів матриці визначатимуть коефіцієнти для L лінійних рівнянь.

Для нашого прикладу з L = 8 і K = 16 представляємо наступний набір рівнянь:

$$\begin{aligned} U_1&=y_1+y_5+y_9+y_{13} & U_5&=y_1+y_2+y_3+y_4\\ U_2&=y_2+y_6+y_{10}+y_{14} & U_6&=y_5+y_6+y_7+y_8\\ U_3&=y_3+y_7+y_{11}+y_{15} & U_7&=y_1+y_8+y_{11}+y_{14}\\ U_4&=y_4+y_8+y_{12}+y_{16} & U_8&=y_2+y_5+y_{12}+y_{15} \end{aligned} \tag{3}$$

Значення y_i розпізнаються за m =8-ма розрядними словами, і після підстановки та обчислення U_j, $j=\overline{1,L}$, L =8 отримаємо L 8 розрядних слів текстового шифру.

Розглянемо операційні витрати на *KeyGen*. Витрати для системи лінійних рівнянь дорівнюють L·K біт. Витрати на відкриті ключі – масиви $(\gamma_{i,j})$ складатимуть mK m-бітних слів. Інші масиви K $(\lambda_{i,j})$ можна побудувати за допомогою генератора випадкових послідовностей. Для прикладу 64-розрядної криптографії (L=8, K=16, m=8) сумарні витрати дорівнюватимуть L·K=128 біт і mK =128 байт.

Реалізація алгоритму Encap

При введенні відкритих ключів-масивів $(\gamma_{i,j})$ алгоритм $(\lambda_{i,j})$ генерує закритий ключ R і зашифровані тексти C_0, C_1. Приватний ключ R складається з двох частин $R=R_1\|R_2$, де R_2 – це випадковий ключ сеансу, призначений для рандомізації C_0, C_1 і відкидається під час декапсуляції. Загальний ключ формується значенням R_1. Зашифровані тексти C_0, C_1 обчислюються в два етапи. Спочатку ми представляємо R у вигляді K m-розрядних слів. Крім того, R_1 виражаємо L у m-розрядних словах r_e, $e=\overline{1,L}$ а R_2 – KL у m-розрядних словах z_e, $e=\overline{1,K-L}$.

Слова r_e є аргументами під час обчислень на масивах $(\gamma_{i,j})_k$ і $(\lambda_{i,j})_k$ – відкритих ключах, побудованих для факторизованих логарифмічних підписів (1), а слова z_e є аргументами під час обчислень на випадкових масивах $(\gamma_{i,j})_k$ і $(\lambda_{i,j})_k$ – нефакторизованих логарифмічних підписах (2). Для обчислених $(\gamma_{i,j}(r_e))_k$ і $(\lambda_{i,j}(r_e))_k$ та обчислених $(\gamma_{i,j}(z_e))_k$ і $(\lambda_{i,j}(z_e))_k$ ми використовуємо порозрядне представлення слів r_j і z_j. Обчислення для аргумента r_e визначаються порозрядним підсумовуванням членів масиву $(\gamma_{i,j})_k$:

$$(\gamma_{i,j}(r_e))_k = (\gamma_{i,j}(r_{e1}, r_{e2}, ..., r_{em}))_k = \sum_{j=1}^{m} (\gamma_{r_{ej},j})_k . \quad (4)$$

Обчислення для масивів $(\lambda_{i,j})_k$ для аргументу r_e також реалізуються за правилами формули (4). Це також вірно для обчисленого за аргументом z_e. Для нашого прикладу 64-розрядної криптографії (L=8, K=16, m=8) табл. 4 представляє згенеровані слова R_1 та R_2.

Таблиця 4

Узагальнювальні слова R_1 і R_2

R_1	R_2
00100011	00011000
00010000	10101010
10101110	10101010
11010100	11111101
00110011	11110000
11111010	11101101
10111101	11010001
00100110	10001001

Продемонструємо обчислення $(\gamma_{i,j}(r_e))_k$ з використанням виразу (4) для першого слова $r_1 = 00100011$з табл. 4. Давайте подивимося на перший масив $(\gamma_{i,j}(r_1))_1$, представлений у табл. 3. Виберіть записи $(\gamma_{i,j}(r_1))_1$ відповідно до бітового представлення r_1.

Коли значення біта дорівнює 0, вибирається перший запис у блоці, що відповідає номеру біта в послідовності, а другий запис вибирається, коли значення біта дорівнює 1. Порядок вибору та обчислений результат представлені у табл. 5. Вибрані записи виділено жирним шрифтом. Блоки підкреслені. Стовпці 3 і 4 показують результати, обчислені для $(\gamma_{i,j}(r_e))_k$, і $k = \overline{1,K}$ – для $(\lambda_{i,j}(r_e))_k$ всіх $(\gamma_{i,j}(z_e))_k$ 16 $(\lambda_{i,j}(z_e))_k$ логарифмічних сигнатур.

Таблиця 5

Обчислені масиви та вибрані записи

$(\gamma_{i,j})_1$	$(\gamma_{i,j}(r_1))_1$	$(\gamma_{i,j}(r_e))_k$, $(\gamma_{i,j}(z_e))_k$	$(\lambda_{i,j}(r_e))_k$ $(\lambda_{i,j}(z_e))_k$
11110111	**11110111+**	01111101	10000110
11010110	**10011011+**	10000111	10001000
10011011	**00001001+**	10110101	00100110
11110101	**00010110+**	01000111	11000011
00011011	**00111000+**	10010000	01000010
00001001	**11110001+**	11101000	01000101
00010110	**00001000+**	00001010	11010101
01110110	**11001111 =**	01101100	10110000
00111000	01111101	10001001	10110001
11101101		11111110	11111010
11110001		00110001	11001010
01001111		10101001	00010100
10111101		10010010	01011001
00001000		01111011	00111011
00000010		11110101	10101000
11001111		01010000	11011010

На другому кроці виконуються розрахунки за лінійними рівняннями (3). У вирази (3) підставимо значення $y_k = (\gamma_{i,j})_k$ та обчислимо L слів шифротексту C_0 . Потім замінюємо $y_k = (\lambda_{i,j})_k$ L слів і отримуємо зашифрований текст C_1. Табл. 6 показує розраховані шифртексти C_0 і C_1 для нашого прикладу. Операційні витрати на передачу зашифрованих текстів складають 2 mL біт.

Таблиця 6

Розраховані шифртексти C_0 і C_1

C_0	C_1
11110110	00101100
11101010	00001100
01111011	10010001
11010010	10111101
00001000	11101011
01011011	11000111
01001011	01110110
00011110	01100010

Реалізація алгоритму Decap

При введенні зашифрованого тексту C_0 , C_1 і закритого ключа sk алгоритм видає загальний ключ ks = R_1 . На першому етапі розшифровки ми повинні розрахувати

$$D_l = C_{0l} + C_{1l}\psi + t_l + \tau_l\psi\,,\ l = \overline{1,L}\,. \qquad (5)$$

Значення C_{0l}, C_{1l}, секретної матриці ψ визначені в табл. 6 і 2. Необхідно розрахувати значення t_l і τ_l .

У табл. 2 представлені масиви секретних параметрів (t_j) та (τ_j) для першої логарифмічної сигнатури ls_1. Для кожної логарифмічної сигнатури $(\tau_j)_k$ необхідно обчислити суму рядків масиву $(t_j)_k$. Для нашого прикладу значення t_k і τ_k представлені в табл. 7.

Далі підставимо значення $y_k = t_k$ і $y_k = \tau_k$ у вираз (3) і обчислимо L слів t_l і τ_l. Отримані значення наведені в графах 3 і 4 табл. 7. Підставляючи C_{0l}, C_{1l}, ψ, t_l і τ_l в (5), отримуємо L m розрядних слів D_l (стовпець 5 табл. 7). Можна показати, що під час обчислення D_l у виразі (5) викреслюються нефакторизовані логарифмічні індекси (2), а L слів D_l будуть з'єднані L лінійними виразами зі значеннями L логарифмічних індексів (1), що розкладаються на множники.

Таблиця 7

Розрахунок значення t_k і τ_k

$t_{1\div16}$	$\tau_{1\div16}$	$t_{1\div8}$	$\tau_{1\div8}$	$D_{1\div8}$	$\beta_{1\div8}$
11101100	10000011	01110011	01100001	00011010	10010100
01010111	10100111	01010111	11101100	11101100	10101011
00100011	10111001	00010110	01100001	01110101	11111110
00101100	00010111	01010011	01001110	00111011	00111011
00111101	11010000	10110100	10001010	11111010	01010001
00010011	10110010	11100001	00111001	10010100	01000111
10111100	10011011	00110000	11001111	11111010	10001011
11001111	10000111	01011101	01111110	10011101	11011111
01011110	11110000				
00001111	00101100				
11011110	11101000				
00001101	00010011				
11111100	11000010				
00011100	11010101				
01010111	10101011				
10111101	11001101				

Ці рівняння випливають із рівнянь (3) із заміною $y_k = \beta_k$ та видаленням y_k відповідних нефакторизованих логарифмічних індексів. Для нашого прикладу отримуємо

$$\begin{aligned} D_1 &= \beta_1 + \beta_5 + \beta_9 & D_5 &= \beta_1 + \beta_2 + \beta_3 + \beta_4 \\ D_2 &= \beta_2 + \beta_6 & D_6 &= \beta_5 + \beta_6 + \beta_7 \\ D_3 &= \beta_3 + \beta_7 & D_7 &= \beta_1 \\ D_4 &= \beta_4 & D_8 &= \beta_2 + \beta_5 \end{aligned} . \quad (6)$$

В шостому стовпці табл. 7 наведено розв'язки невідомих для нашого прикладу β_k.

Останнім кроком дешифрування є факторизація $R_k = \beta_k^{-1}(R_k)$ – відновлення за значенням логарифмічної сигнатури вхідного слова. Перетворення факторизації виконується в порядку, зворотному перетворенню при побудові логарифмічної сигнатури:

$$\beta \xrightarrow{\rho^{-1}_5} \beta_5 \xrightarrow{\rho^{-1}_4} \beta_4 \xrightarrow{\rho^{-1}_3} \beta_3 \xrightarrow{\rho^{-1}_2} \beta_2 \xrightarrow{\rho^{-1}_1} R$$

Перетворення $\rho^{-1}{}_5$ реалізує множення на секретну матрицю $\omega^{-1}{}_{m\times m}$. Перетворення $\rho^{-1}{}_4$ реалізує множення на обернений поліном γ^{-1}. Перетворення $\rho^{-1}{}_3$ усуває шум шляхом додавання s ϖ. Перетворення $\rho^{-1}{}_2$ визначає перестановку бітів у блоках і $\rho^{-1}{}_1$-перестановку блоків. Для значення першої логарифмічної сигнатури $\beta_1 = 10010100$ продемонструємо обчислення на нашому прикладі. Розрахунковий поліном. $\gamma^{-1} = x^2 + x^3 + x^4 + x^5$. Матрицю $\omega^{-1}{}_{m\times m}$ та розрахунки представлено в табл. 8.

Таблиця 8

Матриця $\omega^{-1}{}_{m\times m}$ та розрахунки

$\omega^{-1}{}_{m\times m}$	$\rho^{-1}{}_5 = \beta_1\omega^{-1}{}_{m\times m}$	$\rho^{-1}{}_4 = \beta_5\gamma^{-1}$	SUM (ϖ) $\oplus\beta_4$
01000010	10110000	10001100	00001110
00100110			
01111001			
00101011			
01101011			
11011001			
11000010			
11110011			

Для обчислення $\rho^{-1}{}_3$ обчислимо суму всіх рядків SUM (ϖ)=10000010 матриці ϖ та β_3 = SUM (ϖ) $\oplus\beta_4$.

Розкладання слова β_3 за допомогою простого логарифмічного підпису β_1 з табл. 1 представлено в табл. 9.

Таблиця 9

Приклад β_3 розкладання слова на множники

β_1	Розкладання на множники слова β_3	Результат
0 0000000	0000111 **0**	00011100
10000000	01001110	
00 000000	010000 **0**	
01000000	1011010	
000 00000	11110 **1**	
00100000	001111	
00000000	1100 **1**	
0111 0000	10111	
01000000	011 **1**	
10111 000	0111	
11011000	00 **0**	
001111 00	000	
1011010 0	0 **0**	
01111110	00	
01001110	**0**	
01000111		

Розкладання слова на множники β_3 починається з перегляду нижнього блоку простого логарифмічного підпису β_1.

Перетворення $\rho^{-1}{}_2$= [11011110] визначає перестановку бітів у блоках слова 00011100, отримуємо β_2=11000010. Перетворення $\rho^{-1}{}_1$= [62054371] виконує перестановку бітів у слові 11000010, отримуємо R=00100011. Результат збігається з першим словом ключа R_1 в табл. 4. L значень логарифмічних підписів визначають L факторизацій і відновлення загального ключа R_1.

Висновок

Оцінка обчислювального часу, необхідного для реалізації криптосистеми LINE в протоколі 5G AKA, вимагає розгляду багатьох компонентів протоколу 5G. У [8] проведено часткове порівняння механізмів постквантового спрямованого шифрування – CRYSTALS-Kyber, BIKE, Classic McEliece і HQC – з традиційним ECIES на Curve 25519 і Secp256r1. Автори зосередилися на 128-бітній безпеці, залишивши невирішеною оцінку вартості переходу на 192-бітний і 256-бітний рівні.

Можливі точні прогнози операційних витрат на впровадження. Криптосистема LINE для 128-бітної безпеки може бути реалізована за допомогою 16-бітних слів: m=16, L=8, K=16. Розмір відкритого ключа становитиме 64 байти на логарифмічний підпис, а з урахуванням кількості масивів K=16 загальний розмір відкритого ключа дорівнюватиме 1024 байтам. Усі інші масиви можна згенерувати за допомогою генератора випадкових бітів. Для порівняння, криптосистема CRYSTALS-Kyber вимагає 800 байт, тоді як Classic McEliece вимагає 261120 байт.

Вартість секретних ключів для криптосистеми LINE буде дорівнювати 142 байтам, якщо використовувати однакові секретні перетворення для всіх факторизованих логарифмічних підписів, і 464 байт, якщо використовувати різні перетворення. Для порівняння, CRYSTALS-Kyber вимагає 1632 байти, а Classic McEliece – 6452 байти

Список літератури:

1. Liu F., Peng J., Zuo M. Toward a secure access to 5G network // Proceedings of the 17th IEEE Conference on Trust, Security and Privacy in Computing and Communications (TrustCom '18), New York, NY, USA, August 1–3, 2018. P. 1121–1128.

2. Wang Y., Zhang Z., Xie Y. Privacy-Preserving and Standard-Compatible AKA Protocol for 5G // Proceedings of the 30th USENIX Security Symposium (USENIX Security '21), Online, 11–13 August 2021; USENIX Association: Vancouver, BC, Canada, 2021. P. 3595–3612. Available online: https://www.usenix.org/conference/usenixsecurity21/ presentation/wang-yuchen .

3. Xiao Y., Wu Y. 5G-IPAKA: An improved primary authentication and key agreement protocol for 5g networks // Information. 2022. №13. 125 p.

4. Arkko J., Norrman K., Mattsson JP Forward Secrecy for the Extensible Authentication Protocol Method for Authentication and Key Agreement (EAP-AKA'FS). Internet-Draft draft-ietf-emu-aka-pfs-11, Internet Engineering Task Force. 2023. Available online: https://datatracker.ietf.org/doc/draft-ietf-emu-aka-pfs/11/

5. 3GPP. Security Architecture and Procedures for 5G System TS33.501 v18.2.0. Technical Report, The 3rd Generation Partnership Project. 2023. Available online: https://portal.3gpp.org/desktopmodules/Specifications/ SpecificationDetails.aspx? specificationId=3169.

6. Køien GM The SUCI-AKA Authentication Protocol for 5G Systems. In Proceedings of the 13rd NISK Conference on Norwegian Information Security (NISK'20), Online, 23–25 November 2020. Available online: https://ojs.bibsys.no/index.php/NIK/article/view/885.

7. Xiao Y., Gao S. 5GAKA-LCCO: A secure 5G authentication and key agreement protocol with less communication and computation overhead // Information. 2022. №13. 257 p.

8. Mohamed Taoufiq Damir, Tommi Meskanen, Sara Ramezanian & Valtteri Niemi. A Beyond-5G Authentication and Key Agreement Protocol // International Conference on Network and System Security NSS 2022: Network and System Security 07 December 2022. P. 249–264.

9. 3GPP. Authentication and Key Management for Applications (AKMA) https://portal.3gpp.org/ desktopmodules/ Specifications/ SpecificationDetails.aspx? specificationId=3690

10. NIST. Submission requirements and evaluation criteria for the postquantum cryptography standardization process, 2016. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/ call-for-proposals-final-dec- 2016.

11. Gennady Khalimov, Yevgen Kotukh, Maksym Kolisnyk, Svitlana Khalimova, Oleksandr Sievierinov. LINE: Cryptosystem based on linear equations for logarithmic signatures. https://eprint.iacr.org/2024/697.pdf, Paper 2024/697.

12. Gennady Khalimov, Yevgen Kotukh, Maksym Kolisnyk, Svitlana Khalimova, Oleksandr Sievierinov “SIGNLINE: Digital signature scheme based on linear equations cryptosystem”. https://doi.org/10.48550/arXiv.2405.16227

13. Kotukh Y., & Khalimov H. Advantages of Logarithmic Signatures in the Implementation of Crypto Primitives // Challenges and Issues of Modern Science. 2024. No 2. P. 296–299. https://cims.fti.dp.ua/j/article/view/119

14. Котух Є., Халімов Г., & Коробчинський М. Побудова покращеної схеми шифрування на узагальнених Сузукі 2-групах в криптосистемі MST3 // Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка». 2023. No 2(22). P. 19–30. https://doi.org/10.28925/2663-4023.2023.22.1930

15. Kotukh Y., Khalimov G., Korobchynskyi M., Rudenko M., Liubchak V., Matsyuk S., & Chashchyn M. Research horizons in group cryptography in the context of post-quantum cryptosystems development // Radiotekhnika. 2024. No 216. P. 62–72. https://doi.org/10.30837/rt.2024.1.216.05

16. Gennady Khalimov, Yevgen Kotukh, Oleksandr Sievierinov, Svitlana Khalimova,Sang-Yoon Chang, Yaroslav Balytskyi Strong Encryption Based on the small Ree groups // International Conference “Problems of Infocommunications. Science and Technology” (PIC S&T′2022) 10 – 12 October, 2022 Proceedings, 2022. P. 439–444.

17. Khalimov, G., Kotukh, Y., Chang, S.-Y., Balytskyi, Y. Khalimova, S., Marukhnenko, O. Encryption Scheme Based on the Generalized Suzuki 2-groups and Homomorphic Encryption Communications in Computer and Information Science, 2022, 1536 CCIS. P. 59–76.

18. Khalimov G., Kotukh Y., Khalimova S., ... Marukhnenko O., Tsyplakov D. Towards advance encryption based on a Generalized Suzuki 2-groups // International Conference on Electrical, Computer, Communications and Mechatronics Engineering, ICECCME 2021.

19. Khalimov G., Kotukh Y., Didmanidze I., ... Khalimova S., Vlasov A. Towards three-parameter group encryption scheme for MST3 cryptosystem improvement // Proceedings of the 2021 5th World Conference on Smart Trends in Systems Security and Sustainability. WorldS4 2021, 2021. P. 204–211.

Надійшла до редколегії 03.11.2024

Відомості про авторів:

Котух Євген Володимирович – канд. техн. наук, доцент, професор кафедри кібербезпеки; Національний технічний університет «Дніпровська політехніка»; Дніпро, Україна; e-mail: yevgenkotukh@gmail.com; ORCID: https://orcid.org/0000-0003-4997-620X

Халімов Геннадій Зайдулович – д-р техн. наук, професор, завідувач кафедри безпеки інформаційних технологій; Харківський національний університет радіоелектроніки; Харків, Україна; e-mail: hennadii.khalimov@nure.ua; ORCID: https://orcid.org/0000-0002-2054-9186

Джура Ілля Євгенович – студент 4-го курсу, Національний Авіаційний Університет; Київ, Україна; e-mail: illya773823@gmail.com; ORCID: https://orcid.org/0009-0002-5470-4479

Хівренко Гліб Олександрович – аспірант кафедри безпеки інформаційних технологій; Харківський національний університет радіоелектроніки; Харків, Україна; e-mail: hlib.khivrenko@nure.ua; ORCID: https://orcid.org/0009-0001-7168-1793