

УДК 004.056

Неласа Г.В.¹, Аль-Хамад Н.А.²

¹ канд. техн. наук, доц. НУ «Запорізька політехніка»

² асп. НУ «Запорізька політехніка»

ДОСЛІДЖЕННЯ ТА АНАЛІЗ СХЕМИ ШИФРУВАННЯ ВІДКРИТОГО КЛЮЧА K-PKE В СТАНДАРТІ FIPS 203

Федеральний стандарт обробки інформації (FIPS) 203 запроваджує механізм інкапсуляції ключів на основі модульної решітки (ML-KEM) для захисту даних від загроз квантових обчислень. Великі компанії, включаючи Google, активно впроваджують постквантову криптографію у свої сервіси, зокрема квантово-безпечні цифрові підписи у Cloud Key Management Service [1].

Мета дослідження: дослідити алгоритми, що складають основу ML-KEM (зокрема схему шифрування відкритого ключа K-PKE) та проаналізувати їх функціонування.

Предмет дослідження: стандарт FIPS 203 для побудови безпечного постквантового процесу обміну ключами.

Процес обміну ключами.

В механізмі ML-KEM процес обміну ключами відбувається наступним чином:

Аліса генерує пару ключів – відкритий (інкапсуляції) та секретний (декапсуляції).

Боб, отримавши відкритий ключ Аліси, створює першу копію спільного секретного ключа та зашифрований текст, який асоціюється зі спільним секретним ключем.

Боб надсилає зашифрований текст Алісі.

Аліса дешифрує зашифрований текст і отримує другу копію спільного секретного ключа.

Стандарт FIPS 203 описує спочатку схему шифрування відкритого ключа K-PKE, алгоритми якої використовуються як підпрограми в ML-KEM.

KEM складається з трьох алгоритмів та колекції наборів параметрів.

Три наступні алгоритми зазначені в стандарті [2]:

- 1) імовірний алгоритм генерації ключів, позначений KeyGen;
- 2) імовірний алгоритм "інкапсуляції", позначений Encaps;
- 3) детерміністичний алгоритм "декапсуляції", позначений Decaps.

Набори параметрів.

ML-KEM включає три набори параметрів, що забезпечують різні рівні безпеки: ML-KEM-512 (категорія безпеки 1); ML-KEM-768 (категорія безпеки 3); ML-KEM-1024 (категорія безпеки 5).

У дослідженні використовувались параметри ML-KEM-512 з наступними значеннями: $k = 2$, $\eta_1 = 3$, $\eta_2 = 2$, $du = 10$, $dv = 4$, а також константи $p = 256$ і $q = 3329$.

Алгоритми в схемі K-PKE.

Алгоритм генерації ключів (K-PKE.KeyGen) приймає випадкове значення в якості вхідних даних.

Основні етапи:

1. Генерація псевдовипадкової матриці A та векторів s та e .
2. Перетворення векторів у область ТЧП (теоретико-числове перетворення).
3. Обчислення вектору $t = As + e$.
4. Кодування отриманих векторів t та s для формування ключів.

Результат: згенеровані ключі ek_PKE (800 байт) та dk_PKE (768 байт).

На рис. 1 наведені результати використання алгоритму K-PKE.KeyGen програмою написаною на Python.

```
1 d = os.urandom(32)
2 ek_PKE, dk_PKE = K_PKE_KeyGen(d)
3 print(f"Keys generated:\n{ek_PKE}\n{dk_PKE}\nwith corresponding lengths:\n{len(ek_PKE)} bytes\n{len(dk_PKE)} bytes")
Keys generated:
b'%1G\xd6\xa84\x0ec\xbe\xf1\xcb\x92.1P\xff\xf0\x1aT\xacA\xfb\xf8\xa46\xe9\x11\xe4\x88\xcc\x
b'\xfe7\x15\x93\xd2]B\xca\xc1\x1c\xd7\xb5\x04\xf6E\x86Z\x1b\x15\xc5\x02\xe9\xc75<\xd9\x91\
with corresponding lengths:
800 bytes
768 bytes
```

Рисунок 1 – Результати генерації ключів

Алгоритм шифрування (K-PKE.Encrypt) приймає ключ шифрування, відкритий текст m та випадкове значення в якості вхідних даних.

Основні етапи:

1. Декодування вектора t та регенерація матриці A .
2. Генерація векторів y , $e1$, $e2$ з використанням випадковості.
3. Обчислення векторів u та v , кодування відкритого тексту m у поліном v .
4. Стиснення та кодування результатів.

Результат: зашифрований текст c розміром 768 байт.

На рис. 2 наведені результати використання алгоритму K-PKE.Encrypt програмою написаною на Python.

Алгоритм дешифрування (K-PKE.Decrypt) приймає ключ дешифрування та зашифрований текст в якості вхідних даних.

```
65 m = os.urandom(32)
66 r = os.urandom(32)
67 c = K_PKE_encrypt(ek_PKE, m, r)
68 print(f"generated plaintext message m:\n(m)\nwith corresponding length:\n(len(m)) bytes")
69 print(f"Ciphertext generated:\n(c)\nwith corresponding length:\n(len(c)) bytes")
generated plaintext message m:
b'X\nk\xea\x07\xd9\\\xd9\xf3\xf7\x8f\xd8\xa7\x16\x9be%z)k\xd4q|\xd8\x00\xdb\x81\x1b'!\x01\xcc"
with corresponding length:
32 bytes
ciphertext generated:
b'\xc2\x0f\x00\xed\xae\xf1\x8f\xd2f\x038\x00\xe8\xf96\xee\x0b\xbd\xce\xb7\x16\xb8\x1a\xa9\xf2\x89\xca\x98I64\
with corresponding length:
768 bytes
```

Рисунок 2 – Результати шифрування

Основні етапи:

1. Розпакування та декодування значень u та v .
2. Обчислення справжнього постійного значення v та значення $v' - v$.
3. Декодування відкритого тексту m з полінома v .

Результат: вихідне відкрите повідомлення m' .

На рисунку 3 наведені результати використання алгоритму K-PKE.Decrypt програмою написаною на Python.

```
71 m_ = K_PKE_decrypt(dk_PKE, c)
72 print(f"Decrypted original message:\n{m_}\nwith corresponding length:\n(len(m_)) bytes")
Decrypted original message:
b'X\nk\xea\x07\xd9\\\xd9\xf3\xf7\x8f\xd8\xa7\x16\x9be%z)k\xd4q|\xd8\x00\xdb\x81\x1b'!\x01\xcc"
with corresponding length:
32 bytes
```

Рисунок 3 – Результати дешифрування

З наведених результатів на рис. 2 – 3 видно що результуюче відкрите повідомлення m відповідає оригіналу, тобто $m' = m$. Значить реалізовані алгоритми успішно виконують функції схеми шифрування K-PKE зазначеного в стандарті FIPS 203. Розміри ключів та формат в якому вони представленні також відповідають тим, що зазначені в цьому стандарті.

Висновки: В результаті дослідження були побудовані та проаналізовані алгоритми в схемі K-PKE, що є основою для побудови ML-KEM. Реалізовані алгоритми успішно забезпечують обмін ключами з використанням постквантових криптографічних методів. Результати демонструють, що розшифроване повідомлення точно відповідає оригіналу, що підтверджує коректність та ефективність реалізованих алгоритмів відповідно до специфікацій NIST FIPS 203.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Jennifer Fernick, Andrew Foster (February 21, 2025). “Announcing quantum-safe digital signatures in Cloud KMS”: Google Cloud. [Електронний ресурс]. – Режим доступу: <https://cloud.google.com/blog/products/identity-security/announcing-quantum-safe-digital-signatures-in-cloud-kms>
2. National Institute of Standards and Technology (2024) Module-Lattice-Based Key Encapsulation Mechanism Standard. (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publication (FIPS) NIST FIPS 203. [Електронний ресурс]. – Режим доступу: <https://doi.org/10.6028/NIST.FIPS.203>