Одеський національний університет імені І. І. Мечникова
Факультет математики, фізики та інформаційних технологій
Кафедра алгебри, геометрії та диференціальних рівнянь

Кваліфікаційна робота

на здобуття ступеня вищої освіти «магістр»

«Дослідження рюкзачної криптосистеми та її модифікацій»

«Research of a backpack cryptosystem and its modifications»

Виконала: здобувачка денної форми навчання
спеціальності 111 Математика
Освітня програма «Математика»
Марчук Катерина Володимирівна

Керівник: канд. фіз.-мат. наук, доц. Савастру О. В. _____
Рецензент: канд. фіз.-мат. наук, доц. Бєлозьоров Г.С.

Рекомендовано до захисту:

Протокол засідання кафедри

№ ___ від _____ 2023 р.

Завідувач кафедри

_____ _____

Захищено на засіданні ЕК № _____

Протокол № ___ від _____ 2023 р.

Оцінка _____ / _____ / _____

Голова ЕК

_____ _____

Одеса — 2023 р.

ЗМІСТ

ВСТУП

Кваліфікаційна робота присвячена дослідженню рюкзачної криптосистеми та її модифікацій.

Актуальність. Для запобігання використанню переданих даних застосовується криптографія для перетворення відкритого тексту в зашифрований. Одним з видів криптографії є криптосистеми з відкритим ключем, які використовують два різні ключі (відкритий і закритий) для шифрування і дешифрування даних.

Однією з перших криптосистем з відкритим ключем є рюкзачна криптосистема Меркла—Хеллмана, яка була винайдена Ральфом Ерклом та Мартіном Хеллманом у 1978 році і заснована на використанні "Задачі про суму підмножин". Використання цієї задачі у рюкзачній криптосистемі Меркла-Хеллмана повинно було зробити її складною і важкою для злому, однак у 1982 році Аді Шамір зламав її. Тому було проведено декілька досліджень для покращення безпеки цієї криптосистеми. Як результат, було розроблено багато модифікацій, які виявились стійкими до більшості атак.

Мета і задачі роботи

Метою роботи є дослідження рюкзачної криптосистеми та її модифікацій, встановлення їхніх переваг та недоліків. Досягненню мети сприяють такі основні задачі роботи.

1) Охарактеризувати стандартну рюкзачну криптосистему.
2) Описати алгоритми генерації ключів, шифрування і дешифрування.
3) Встановити її переваги та недоліки, зокрема, описати алгоритми злому.
4) Дослідити модифікації криптосистеми та визначити їхні переваги над стандартною рюкзачною криптосистемою.
5) Практично реалізувати криптосистему.

Структура роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків і списку літературу.

У першому розділі описано задачу про рюкзак, наведено алгоритми генерації ключів, шифрування і дешифруваня повідомлення для стандартної рюкзачної криптосистеми і приклади їх застосування, перераховано основні

переваги та недоліки криптосистеми, а також можливі атаки для її злому.

Другий розділ присвячено дослідженню модифікацій стандартної рюкзачної криптосистеми.

У третьому розділі наведено практичну реалізацію криптосистеми.

РОЗДІЛ 1

СТАНДАРТНА РЮКЗАЧНА КРИПТОСИСТЕМА МЕРКЛА-ХЕЛЛМАНА

1.1 Загальна характеристика криптосистем з відкритим ключем

Ідея криптографії з відкритим ключем пов'язана з ідеєю односторонніх функцій. Адже за заданим аргументом x легко обчислити значення функції $f(x)$, тоді як навпаки, значення x з $f(x)$, знайти важко. Ця ситуація зображена на рисунку 1.1.

$$x \underset{\text{важко обчислити}}{\overset{\text{легко обчислити}}{\rightleftarrows}} f(x)$$

Рис. 1.1

У 1976 році Діффі і Хеллман у своїй роботі «Нові напрямки в криптографії» [4] запропонували використовувати для шифрування односторонню функцію з лазівкою.

Одностороння функція — це функція $f\colon A \longrightarrow B$ з такими властивостями:

1) $f(a)$ легко обчислити для будь-якого $a \in A$.
2) неможливо обчислити $f^{\leftarrow}(b)$ майже для всіх $b \in B$.

Одностороння функція з лазівкою — це одностороння функція з ще однією властивістю

3) $f^{\leftarrow}(b)$, $b \in B$, легко обчислити, якщо відома деяка додаткова інформація.

Властивість 1) робить таку функцію практичною в застосуванні, тоді як властивість 2) забезпечує безпеку при використанні f з метою шифрува-

ння. А остання властивість робить можливим дешифрування повідомлення отримувачем.

У криптосистемах з відкритим ключем використовуються два ключі: відкритий (публічний) та закритий (приватний).

Відкритий ключ може бути опублікованим для загального доступу, тому будь-хто за допомогою цього ключа може зашифрувати своє повідомлення. А розшифрувати надіслане повідомлення зможе тільки той, у кого є закритий ключ.

Таким чином, суть таких криптосистем полягає в односторонньому математичному зв'язку, що існує між двома ключами: інформація про відкритий ключ ніяк не допоможе відновити закритий, але володіння закритим ключем забезпечує можливість розшифровувати повідомлення, які були зашифровані з використанням відкритого.

Підвищена безпека даних, яку забезпечує криптографія з відкритим ключем, є її основною перевагою. Розглянемо будь-яку класичну (симетричну) криптосистему. Ключ шифрування дає також і ключ розшифрування. Це означає, що відправник і одержувач домовляються заздалегідь про алгоритм шифрування. Це можливо зробити або під час особистої зустрічі, або під час передачі ключа шифрування по секретному каналу. У разі використання криптосистеми з відкритим ключем обидві сторони не зустрічаються особисто, вони навіть можуть не знати одне одного і використовувати будь-які види зв'язку. Користувачам не потрібно нікому передавати свої приватні ключі, а це зменшує шанси виявлення сторонніми людьми секретного ключа під час передачі.

Як правило, шифрування з відкритим ключем є кращим методом криптографії, коли існує багатокористувацьке середовище і необхідно забезпечити конфіденційність за допомогою розподілу ключів і цифрових підписів для перевірки ідентичності користувачів.

У таблиці 1.1 наведено основні переваги криптосистем з відкритим ключем над класичними (симетричними).

Характеристика	Симетричні	З відкритим ключем
Ключ	Один ключ для шифрування і дешифрування	Наявність відкритого і закритого ключів
Обмін ключами	Потрібен секретний канал або особиста зустріч	Відкритий ключ доступний усім
Математична складність	Відносно прості математичні операції	Складні математичні обчислення
Криптографічна стійкість	Задовільна	Достатня
Вид захисту	Конфіденційність	Конфіденційність, цілісність, автентичність

Табл. 1.1. Порівняння симетричних і криптосистем з відкритим ключем

1.2 Задача про рюкзак

Рюкзачна криптосистема заснована на проблемі суми підмножин (або задачі про рюкзак). Простими словами її можна сформулювати так: є предмети різної маси, чи можна деякі з них покласти у рюкзак так, щоб вага рюкзака дорівнювала певному значенню? Тобто, дано значення (сума) V і набір значень $s_1, s_2, \ldots, s_n$. Необхідно знайти такі $\varepsilon_i \in [0; 1]$, щоб виконувалась рівність

$$V = s_1\varepsilon_1 + s_2\varepsilon_2 + \cdots + s_n\varepsilon_n.$$

Приклад 1.1. Дано значення $s_1 = 2$, $s_2 = 3$, $s_3 = 5$, $s_4 = 15$, $s_5 = 21$.

У таблиці 1.2 наведено розв'язок цієї задачі для декількох випадків.

Відкритий текст	10011	11010	01100	00000
Рюкзак	2 3 7 13 20	2 3 7 13 20	2 3 7 13 20	2 3 7 13 20
Зашифрований текст	2+13+20=35	2+3+13=18	3+7=10	0=0

Табл. 1.2. Розв'язок задачі про рюкзак

Очевидно, що така задача може мати декілька розв'язків або не мати взагалі. Дійсно, у прикладі 1.1 рівняння

$$2\varepsilon_1 + 3\varepsilon_2 + 7\varepsilon_3 + 13\varepsilon_4 + 20\varepsilon_5 = 19$$

не має розв'язків, а рівняння

$$2\varepsilon_1 + 3\varepsilon_2 + 7\varepsilon_3 + 13\varepsilon_4 + 20\varepsilon_5 = 23$$

має два ровз'язки, а саме

$$\varepsilon_1 = \varepsilon 5 = 0,\ \varepsilon_2 = \varepsilon_3 = \varepsilon_4 = 1 \text{ і } \varepsilon_1 = \varepsilon_3 = \varepsilon_4 = 0,\ \varepsilon_2 = \varepsilon_5 = 1.$$

Тому ідея полягає в тому, аби створити дві задачі про рюкзак: просту (яку легко розв'язати) і складну. Для того, щоб описати їх, необхідно ввести додаткове означення.

Означення 1.1. Суперзростаюча послідовність — це послідовність $S_n = \{s_n\}_{n=0}^{N-1}$ додатніх чисел таких, що $s_i > \sum_{j=0}^{i-1} s_j$ для усіх i, $0 \leq i \leq N-1$.

Отже, якщо перелік мас предметів є суперзростаючою послідовністю, то задачу про рюкзак легко розв'зяти. Якщо ж перелік мас предметів є нормальною послідовністю, то таку задачу складно розв'язати.

Особливістю розв'язків задач з суперзростаючою послідовністю є їхня єдиність. Доведемо це.

Теорема 1.1. *Нехай $\{s_i\}_{i=1}^{n} \in \mathbb{N}$ — суперзростаюча послідовність. Якщо $\varepsilon_i, \delta_i \in [0;1]$ і*

$$\sum_{i=1}^{n} \varepsilon_i s_i = \sum_{i=1}^{n} \delta_i s_i,$$

тоді $\varepsilon_i = \delta_i$ для усіх i.

Доведення. Застосуємо індукцію по n. Якщо $n = 1$, то це тривіальний випадок, оскільки з $\varepsilon_i s_i = \delta_i s_i$ випливає, що $\varepsilon_i = \delta_i$.

Припустимо, що $n > 1$. Тоді з рівності $\sum\limits_{i \leq n} \varepsilon_i s_i = \sum\limits_{i \leq n} \delta_i s_i$ випливає, що

$$|\varepsilon_n - \delta_n| a_n = \left| \sum_{i<n} (\delta_n - \varepsilon_n) a_i \right| \leq \sum_{i<n} a_i < a_n.$$

Оскільки, $|\varepsilon_n - \delta_n| \leq 1$, то $\varepsilon_n = \delta_n$. Таким чином,

$$\sum_{i \leq n} \varepsilon_i s_i = \sum_{i \leq n} \delta_i s_i,$$

і $\varepsilon_i = \delta_i$ для $i < n$ за припущенням індукції.

Тому $\varepsilon_i = \delta_i$ для всіх $i \leq n$. □

Алгоритм розв'язку легкої задачі про рюкзак.

1) Порівняти вагу рюкзака з найбільшим числом в послідовності. Якщо це число більше за повну вагу, то його не треба класти в рюкзак.
2) Якщо це число менше або дорівнює повній вазі, то покласти його в рюкзак. Вагу рюкзака зменшити на це число.
3) Перейти до наступного за величиною числа цієї послідовності.
4) Повторити цю процедуру з усіма числами послідовності. Якщо повна вага рюкзака зменшиться до нуля, то розв'язок знайдено.

Приклад 1.2. Дано суперзростаюча послідовність $S = \{3, 4, 6, 15, 28, 56\}$ і вага рюкзака $V = 75$. Знайти розв'язок задачі.

- Найбільше число послідовності — 56. Повна вага рюкзака більша за це число, отже, кладемо його у рюкзак. Нова вага рюкзака: $75 - 56 = 19$.
- Перейдемо до числа 28. $28 > 19$, отже, його не кладемо у рюкзак.
- Перейдемо до числа 15. Це число менше за вагу рюкзака, отже, його кладемо туди. Тоді нова вага буде $19 - 15 = 4$.
- Число 6 більше за вагу рюкзака, тому його не кладемо.
- Число 4 збігається з вагою рюкзака, тому його кладемо, а нова вага буде $4 - 4 = 0$.

Оскільки отримали 0, то ровз'язок знайдено, тому відкритий текст матиме вигляд 010101.

Рюкзачна криптосистема Меркла—Хеллмана побудована таким чином: закритий ключ є суперзростаючою послідовністю легкої задачі про рюкзак, а відкритий — послідовністю складної задачі з тим самим розв'язком. Далі детальніше розглянемо роботу алгоритмів цієї криптосистеми.

1.3 Процес генерації ключів

Наведемо алгоритм для процесу генерації ключів.

1) Обрати суперзростаючу послідовність $S = \{s_i\}_{i=0}^{k}$.
2) Обрати ціле число n таке, що $n \geq \sum_i^k s_i$.
3) Обрати число u таке, що $(u, n) = 1$. Таким чином, (S, n, u) — це приватний ключ, який зберігається в секреті.
4) Обчислити $q_i = u \cdot s_i \pmod{n}$, де $1 \leq i \leq k$. Тоді послідовність $Q = \{q_i\}_{i=1}^{k}$ — це відкритий ключ, який доступний кожному.

Приклад 1.3. Згенерувати приватний та публічний ключі для обраної послідовності.

Оберемо суперзростаючу послідовність $S = \{2, 3, 6, 13, 27, 52\}$. Тоді $\sum_i^k s_i = 103$. Оберемо числа $n = 105$ і $u = 31$ так, що $(105, 31) = 1$. Отже, $(\{2, 3, 6, 13, 27, 52\}, 105, 31)$ — приватний ключ.

Обчислимо $q_i = 31 \cdot s_i \pmod{105}$, де $1 \leq i \leq 6$.

$q_1 = 31 \cdot 2 \pmod{105} = 62 \pmod{105}$;

$q_2 = 31 \cdot 3 \pmod{105} = 93 \pmod{105}$;

$q_3 = 31 \cdot 6 \pmod{105} = 186 \pmod{105} = 81 \pmod{105}$;

$q_4 = 31 \cdot 13 \pmod{105} = 403 \pmod{105} = 88 \pmod{105}$;

$q_5 = 31 \cdot 27 \pmod{105} = 837 \pmod{105} = 102 \pmod{105}$;

$q_6 = 31 \cdot 52 \pmod{105} = 1612 \pmod{105} = 37 \pmod{105}$.

Отже, $Q = \{62, 93, 81, 88, 102, 37\}$ — публічний ключ.

1.4 Процес шифрування

Наведемо алгоритм для процесу шифрування повідомлення.

1) Перетворити кожний символ відкритого тексту у двійкову форму b_i довжиною k біт, де $1 \leq i \leq k$, і записати послідовність $B = \{b_i\}_{i=0}^{k}$.

2) Для кожного b_i обчислити відповідний вираз en_i, де

$$en_i = \sum_{j=1}^{k} q_j \cdot b_{ij}.$$

Тоді $En = \{en_i\}_{i=1}^{k}$ — зашифрований текст.

Приклад 1.4. Зашифрувати слово MATH, використовуючи публічний ключ, отриманий в прикладі 1.3.

$Q = \{62, 93, 81, 88, 102, 37\}$ — публічний ключ. Перетворимо кожний символ тексту у двійкову форму довжини $k = 6$.

M — 001101

A — 000001

T — 010100

H — 001000

Отже, отримали послідовність $B = \{001101, 000001, 010100, 001000\}$.

Для кожного члена послідовності B розрахуємо значення en_i за формулою

$$en_i = \sum_{j=1}^{k} q_j \cdot b_{ij}.$$

.

$en_1 = 0 \cdot 62 + 0 \cdot 93 + 1 \cdot 81 + 1 \cdot 88 + 0 \cdot 102 + 1 \cdot 37 = 206;$

$en_2 = 0 \cdot 62 + 0 \cdot 93 + 0 \cdot 81 + 0 \cdot 88 + 0 \cdot 102 + 1 \cdot 37 = 37;$

$en_3 = 0 \cdot 62 + 1 \cdot 93 + 0 \cdot 81 + 1 \cdot 88 + 0 \cdot 102 + 0 \cdot 37 = 181;$

$en_4 = 0 \cdot 62 + 0 \cdot 93 + 1 \cdot 81 + 0 \cdot 88 + 0 \cdot 102 + 0 \cdot 37 = 81.$

Тоді послідовність $En = \{206, 37, 181, 81\}$ — зашифрований текст.

1.5 Процес дешифрування

Для цього процесу необхідно знати закритий ключ (S, n, u). Алгоритм процесу дешифруваня виглядає таким чином:

1) За допомогою розширеного алгоритма Евкліда знайти u^{-1} таке, що $uu^{-1} = 1 \pmod{n}$.
2) Обчислити значення l_i за формулою

$$l_i = en_i \cdot u^{-1} \pmod{n} = \left(\sum_{j=1}^{k} q_j \cdot b_{ij} \right) \cdot u^{-1} \pmod{n}.$$

3) Відняти найбільше число в S від l_i. Повторювати цей процес допоки в результаті не буде 0. Отримання нуля означає, що утворилось b_i, яке представляє собою двійкову формулу для i-го символу у відкритому тексті.

Теорема 1.2. *Довести, що алгоритм дешифрування дійсно дозволяє відновити оригінальний відкритий текст (повідомлення).*

Доведення. Дійсно,

$$l_i \equiv u^{-1} \cdot en_i \equiv u^{-1} \sum_{j=1}^{k} q_j \cdot b_{ij} \equiv \sum_{j=1}^{k} b_j \cdot s_j \pmod{n}.$$

Оскільки, $0 \leq l_i \leq n$, $l_i = \sum_{j=1}^{k} b_j \cdot s_j \pmod{n}$, і отже розв'язок легкої задачі про рюкзак на кроці 3) дає біти вихідного повідомлення. □

Приклад 1.5. Розшифрувати текст $En = \{206, 37, 181, 81\}$ за допомогою приватного ключа, отриманого в прикладі 1.3.

($\{2, 3, 6, 13, 27, 52\}$, 105, 31) — приватний ключ.

Обчислимо u^{-1} таке, що $31 \cdot u^{-1} = 1 \pmod{105}$. Отже, $u^{-1} = 61$.

Далі обчислимо значення l_i за формулою

$$l_i = en_i \cdot 61 \pmod{105}.$$

$l_1 = 206 \cdot 61 \pmod{105} = 12566 \pmod{105} = 71 \pmod{105}$;

$l_2 = 37 \cdot 61 \pmod{105} = 2257 \pmod{105} = 52 \pmod{105}$;

$l_3 = 181 \cdot 61 \pmod{105} = 11041 \pmod{105} = 16 \pmod{105}$;

$l_4 = 81 \cdot 61 \pmod{105} = 4941 \pmod{105} = 6 \pmod{105}$.

Далі необхідно знайти двійкове представлення для b_i.

$71 - 52 = 19 - 13 = 6 - 6 = 0$. Отже, $b_1 = 001101$.

$52 - 52 = 0$. Отже, $b_2 = 000001$.

$16 - 13 = 3 - 3 = 0$. Отже, $b_3 = 010100$.

$6 - 6 = 0$. Отже, $b_4 = 001000$.

$B = \{0011001, 000001, 010100, 001000\}$ — розшифрований текст у двійковій формі. Тоді, було зашифроване слово MATH.

1.6 Переваги та недоліки

У рюкзачної криптосистеми Меркла—Хеллмана є свої переваги і недоліки, які ми розглянемо детальніше нижче.

Переваги.

Однією з основних переваг є процес генерації ключів. Тому що для цього процесу потрібна суперзростаюча послідовність, яку важко вгадати чи відтворити. Більш того, кожне повідомлення має свій приватний ключ, що підвищує безпеку, адже запобігає застосуванню атак відкритого тексту. Так само атаки грубої сили є недоцільними через великий простір чисел, адже це вимагає величезних обчислювальних потужностей і багато часу.

Ще однією перевагою є швидкість. Ці системи досягають дуже високих швидкостей шифрування і дешифрування. Крім того, рюкзачне шифрування забезпечує високий рівень конфіденційності повідомлень, оскільки шифрування одного перекидання бітів у вихідному повідомленні призводить до випадкової зміни більше половини зашифрованих бітів.

Недоліки.

Однією з потенційних слабких сторін є те, що якщо зловмисник зможе визначити підмножину суперзростаючої послідовності, яка використовується для генерації відкритого ключа, він може зламати систему шифрування. Більш того, використання слабких генераторів випадкових чисел також збільшує шанси злому.

Основним недоліком є її вразливість до певних типів атак, а саме атаки Шаміра [11] та застосування алгоритму LLL (Ленстри — Ленстри – Ловаса). Розглянемо детальніше алгоритм злому рюкзачної криптосистеми за допомогою алгоритму LLL.

Припустимо, що є публічний ключ $Q = \{q_1, q_2, \ldots, q_k\}$ і блок тексту en, який був зашифрований за допомогою цього ключа. Якщо зловмисник знає ключ Q і текст en, то він може зламати систему, розв'язавши матричне рівняння $TU = en$, де U — це матриця розміру $k \times 1$, яка скадається з нулів та одиниць.

Для цього необхідно застосувати алгоритм LLL до матриці

$$M = \begin{bmatrix} I_{k\times k} & 0_{k\times 1} \\ Q_{1\times k} & -en_i \end{bmatrix}_{k+1\times k+1}, 1 \le i \le k$$

де $I_{k\times k}$ — одинична матриця, $Q_{1\times k}$ — публічний ключ, а en_i — i-тий елемент зашифрованого тексту.

Приклад 1.6. [7] Дано публічний ключ $Q = \{82, 123, 287, 83, 248, 373, 10, 471\}$. Необхідо зашифрувати повідомлення $M = 10010110$. Тоді зашифрований текст буде

$$en = 1 \cdot 82 + 0 \cdot 123 + 0 \cdot 287 + 1 \cdot 83 + 0 \cdot 248 + 1 \cdot 373 + 1 \cdot 10 + 0 \cdot 471 = 548.$$

Тепер припустимо, що нам необхідно відновити відкритий текст, який відповідає зашифрованому тексту $en = 548$. Тоді нам треба застосувати алгоритм LLL до такої матриці

$$M = \begin{bmatrix} I_{8\times 8} & 0_{8\times 1} \\ Q_{1\times 8} & -en_{i_{1\times 1}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 82 & 123 & 287 & 83 & 248 & 373 & 10 & 471 & -548 \end{bmatrix}$$

Було застосовано функцію LatticeReduce[matrix] в Wolfram Mathematica для автоматизації підрахунків.

Отже, алгоритм LLL вивів матрицю M'

$$M' = \begin{bmatrix} -1 & -1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & -1 & 1 & 2 \\ 1 & -1 & -1 & 1 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & -2 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 \\ 1 & -1 & 1 & 0 & 0 & 1 & -1 & 2 & 0 \end{bmatrix}$$

Четвертий рядок має правильний запис, щоб бути розв'язком. Отже, отримали ймовірний розв'язок $U = (1, 0, 0, 1, 0, 1, 1, 0)$. Використовуючи

публічний ключ і зашифрований текст, можна встановити, що знайдений розв'язок дійсно є оригінальним текстом.

Для усунення вразливостей було проводено багато досліджень з метою вдосконалення рюкзачної криптосистеми Меркла—Хеллмана. У наступному розділі буде розглянуто декілька з них.

РОЗДІЛ 2

МОДИФІКАЦІЇ СТАНДАРТНОЇ РЮКЗАЧНОЇ КРИПТОСИСТЕМИ

2.1 Рюкзачна криптосистема з використанням символу Лежандра

Спочатку наведемо відомі означення, які будуть необхідні для характеристики цієї криптосистеми.

Означення 2.1. Нехай a — ціле число, p — просте непарне число. a є квадратичним лишком за модулем p, якщо $(a, p) = 1$ і конгруенція $x^2 \equiv a \pmod{p}$ має розв'язок. Інакше, a — квадратичний нелишок.

Означення 2.2. Символ Лежандра $\left(\frac{a}{p}\right)$ визначається таким чином:

$$\left(\frac{a}{p}\right) = \begin{cases} 1, & \text{якщо a - квадратичний лишок;} \\ -1, & \text{якщо a - квадратичний нелишок.} \end{cases}$$

Отже, одна з модифікації стандартної рюкзачної крипптосистеми заснована на значенні символу Лежандра. Спочатку обирається велике просте число p. Потім обчислюються квадратичні лишки і квадратичні нелишки a за модулем p і сортуються випадковим чином у множині. Оскільки символ Лежандра дорівнює або 1, або -1, для генерації ключів використовуються два окремих процеси, а не один, як у стандартній рюкзачній криптосистемі. Тобто,

$$\left(\frac{a}{p}\right) = \begin{cases} 1, & \text{процес генерації ключів 1;} \\ -1, & \text{процес генерації ключів 2.} \end{cases}$$

Розглянемо детальніше запропонований алгоритм [5].

Процес генерації ключів.

1) Згенерувати послідовність S_1, обрати числа n_1, u_1 такі, що $(u_1, n_1) = 1$. Тоді (S_1, n_1, u_1) — це перший закритий ключ. Користуючись формулою $q_i = u_1 * s_i \ (mod\ n)$, отримаємо послідовність $Q_1 = \{q_i\}_{i=1}^{k}$, яка є першим відкритим ключем.
2) Повторити процес для послідовності S_2 і чисел n_2 , u_2. Тоді (S_2, n_2, u_2) — це другий закритий ключ, а $Q_2 = \{q_i\}_{i=1}^{k}$ — другий відкритий ключ.

Приклад 2.1. Згенерувати публічні та приватні ключі для обраних послідовностей.

Для початку оберемо $p = 19$. Тоді відсортована випадковим чином множина квадратичних лишків і нелишків матиме вигляд $\{4, 2, 7, 3, 8, 13, 5, \dots\}$. Обчислимо відповідні символи Лежандра.

$$\left(\frac{4}{19}\right) = \left(\frac{2^2}{19}\right) = 1;$$

$$\left(\frac{2}{19}\right) = (-1)^{\frac{19^2-1}{8}} = -1;$$

$$\left(\frac{7}{19}\right) \equiv 7^{\frac{19-1}{2}} \pmod{19} = 1;$$

$$\left(\frac{3}{19}\right) \equiv 3^{\frac{19-1}{2}} \pmod{19} = -1;$$

$$\left(\frac{8}{19}\right) = \left(\frac{2}{19}\right) \cdot \left(\frac{4}{19}\right) = -1 \cdot 1 = -1;$$

$$\left(\frac{13}{19}\right) \equiv 13^{\frac{19-1}{2}} \pmod{19} = -1;$$

$$\left(\frac{5}{19}\right) \equiv 5^{\frac{19-1}{2}} \pmod{19} = 1.$$

Отже, множина символів Лежандра — $\{1, -1, 1, -1, -1, -1, 1, \dots\}$.

Процес 1.

Оберемо послідовність $S_1 = \{3, 5, 11, 20, 41\}$, числа $n_1 = 75$ і $u_1 = 46$. Тоді, $(\{3, 5, 11, 20, 41\}, 75, 46)$ — перший приватний ключ.

Обчислимо q_i за формулою $q_i = 46 \cdot s_i \pmod{75}$:

$q_1 = 46 \cdot 3 \pmod{75} = 138 \pmod{75} = 63;$

$q_2 = 46 \cdot 5 \pmod{75} = 230 \pmod{75} = 5;$

$q_3 = 46 \cdot 11 \pmod{75} = 506 \pmod{75} = 56;$

$q_4 = 46 \cdot 20 \pmod{75} = 920 \pmod{75} = 20;$

$q_5 = 46 \cdot 41 \pmod{75} = 1886 \pmod{75} = 11.$

Тоді, $Q_1 = \{63, 5, 56, 20, 11\}$ — перший публічний ключ.

Процес 2.

Аналогічно оберемо послідовність $S_2 = \{2, 3, 7, 13, 27\}$, числа $n_2 = 50$ і $u_2 = 3$. Тоді, $(\{2, 3, 7, 13, 27\}, 50, 3)$ — другий приватний ключ.

Обчислимо q_i за формулою $q_i = 3 \cdot s_i \pmod{50}$:

$q_1 = 3 \cdot 2 \pmod{50} = 6;$

$q_2 = 3 \cdot 3 \pmod{50} = 9;$

$q_3 = 3 \cdot 7 \pmod{50} = 21;$

$q_4 = 3 \cdot 13 \pmod{50} = 39;$

$q_5 = 3 \cdot 27 \pmod{50} = 81 \pmod{50} = 31.$

$Q_2 = \{6, 9, 21, 39, 31\}$ — другий публічний ключ.

Процес шифрування.

1) Перетворити відкритий текст у двійкову форму.
2) Обчислити $en_i = \sum_{j=1}^{k} q_i \cdot b_{ij}$ на основі отриманого відкритого ключа, враховуючи узгоджений набір символу Лежандра. $En = \{en_i\}_{i=1}^{k}$ — зашифрований текст.

Приклад 2.2. Зашифрувати слово MATH, використовуючи публічні ключі, отримані в прикладі 2.1.

Спочатку конвертуємо слово у двійкову форму і оберемо публічний ключ, враховуючи значення символу Лежандра. Цей процес показано у таблиці 2.1.

Обчислимо тепер en_i за формулою $en_i = \sum_{j=1}^{k} q_i \cdot b_{ij}$:

$en_1 = 0 \cdot 63 + 1 \cdot 5 + 1 \cdot 56 + 0 \cdot 20 + 1 \cdot 11 = 72;$

$en_2 = 0 \cdot 6 + 0 \cdot 9 + 0 \cdot 21 + 0 \cdot 39 + 1 \cdot 31 = 31;$

$en_3 = 1 \cdot 63 + 0 \cdot 5 + 1 \cdot 56 + 0 \cdot 20 + 0 \cdot 11 = 119;$

$en_4 = 0 \cdot 6 + 1 \cdot 9 + 0 \cdot 21 + 0 \cdot 39 + 0 \cdot 31 = 9.$

Символи слова	b_i	$\left(\frac{a}{p}\right)$	Ключ, який використовується
M	01101	1	$Q_1 = \{63, 5, 56, 20, 11\}$
A	00001	-1	$Q_2 = \{6, 9, 21, 39, 31\}$
T	10100	1	$Q_1 = \{63, 5, 56, 20, 11\}$
H	01000	-1	$Q_2 = \{6, 9, 21, 39, 31\}$

Табл. 2.1. Процес шифрування

Отже, $En = \{72, 31, 119, 9\}$ — зашифрований текст.

Процес дешифрування.

1) Обчислити оберенені елементи для u_1 за модулем n_1 і u_2 за модулем n_2.
2) Враховуючи набір символу Лежандра, порахувати значення $l_i = en_i \cdot u_j^{-1} \pmod{n_j}$
3) Обчислити b_i, віднімаючи найбільший член в S_i від l_i. Повторити процес з іншими членами S_i. Тоді $B = \{b_i\}_{i=1}^{k}$ — послідовність двійкових представлень i-го символу відкритого тексту.

Приклад 2.3. Розшифрувати текст $En = \{72, 31, 119, 9\}$, використовуючи ключі, отримані в прикладі 2.1.

Знайдемо u_1^{-1}, u_2^{-1} такі, що $u_1^{-1} \cdot 46 = 1 \pmod{75}$, $u_2^{-1} \cdot 3 = 1 \pmod{50}$. Тоді, $u_1^{-1} = 31$ і $u_2^{-1} = 17$. У таблиці 2.2 показано, як буде відбуватись процес дешифрування, враховуючи значення символа Лежандра.

Далі обчислимо значення l_i за формулою $l_i = en_i \cdot u_j^{-1} \pmod{n_j}$, де $1 \leq j \leq 2$, $1 \leq i \leq 4$.

$l_1 = 72 \cdot 31 \pmod{75} = 2232 \pmod{75} = 57;$

$l_2 = 31 \cdot 17 \pmod{50} = 527 \pmod{75} = 27;$

$l_3 = 119 \cdot 31 \pmod{75} = 3689 \pmod{75} = 14;$

en_i	$\left(\frac{a}{p}\right)$	Процес генерації ключів, який використовується	n_i	u_i^{-1}
72	1	1	75	31
31	-1	2	50	17
119	1	1	75	31
9	-1	2	50	17

Табл. 2.2. Процес дешифрування

$l_4 = 9 \cdot 17 \pmod{50} = 153 \pmod{75} = 3.$

Залишилось знайти двійкове представлення для b_i.

$57 - 41 = 16 - 11 = 5 - 5 = 0$ Отже, $b_1 = 01101$.

$27 - 27 = 0$ Отже, $b_2 = 00001$.

$14 - 11 = 3 - 3 = 0$ Отже, $b_3 = 10100$.

$3 - 3 = 0$ Отже, $b_4 = 01000$.

$B = \{01101, 00001, 10100, 01000\}$ — розшифрований текст. Тоді, було зашифровано слово МАТН.

Наведемо переваги цієї модифікації над стандарнтою рюкзачною криптосистемою.

1) Алгоритм LLL не може бути застосований, щоб зламати запропоновану криптосистему, оскільки існує два різних процеси генерації ключів. Ці процеси використовують дві різні суперзростаючі послідовності довжини k для генерації двох різних відкритих ключів. Тому використання будь-якого відкритого ключа довжини разом із зашифрованим текстом не допоможе зламувачу відновити відкритий текст.
2) Також атака Шаміра не може бути серйозним ризиком для запропонованої криптосистеми. Оскільки вона використовує два різних відкритих ключі на основі символу Лежандра.
3) Порівняння запропонованої та стандартної криптосистем показує, що процес шифрування та дешифрування займає більше часу порівняно зі стандартною криптосистемою. Збільшення часу дешифрування означає більше часу на злам системи, якщо можуть бути застосовані

будь-які інші можливі атаки.

Таким чином, результати показують, що запропонована криптосистема є безпечнішою та більш ефективною у порівнянні зі стандартною криптосистемою.

2.2 Рюкзачна криптосистема Шора — Рівеста

Наразі вона є єдиною відомою схемою шифрування, заснованою на задачі про рюзкак, яка не використовує модульного множення для маскування простої задачі про рюкзак.

Процес генерації ключів.

1) Обрати скінченне поле $\mathbb{F}_q$ характеристики p, де $q = p^h$, $p \geq h$, і для якого розв'язується задача дискретного логарифмування.
2) Обрати будь-який незвідний многочлен $f(x)$ степеня h над $\mathbb{Z}_p$. Елементи $\mathbb{F}_q$ будуть представлені у вигляді многочленів в $\mathbb{Z}_p$ степеня меншого за h, з множенням за модулем $f(x)$.
3) Обрати будь-який примітивний елемент $g(x)$ поля $\mathbb{F}_q$.
4) Для кожного елемента основного поля $i \in \mathbb{Z}_p$, знайти дискретний логарифм $a_i = \log_{g(x)}(x + i)$ елемента поля $(x + i)$ з основною $g(x)$.
5) Обрати будь-яке ціле число d, $0 \leq d \leq p^h - 2$.
6) Обчислити $c_i = (a_i + d) \pmod{p^h - 1}$, $0 \leq i \leq p - 1$.
7) Тоді, публічний ключ — це $(\{c_0, c_1, \ldots, c_{p-1}\}, p, h)$, а приватний ключ — $(f(x), g(x), d)$.

Приклад 2.4. [8] Згенерувати публічний та приватний ключі, враховуючи, що $p = 7$, $h = 4$.

Оберемо незвідний многочлен $f(x) = x^4 + 3x^3 + 5x^2 + 6x + 2$ степеня 4 над $\mathbb{Z}_7$. Тоді елементи скінченного поля $\mathbb{F}_{7^4}$ представлені як многочлени в $\mathbb{Z}_7[x]$ степеня, меншого за 4, з множенням за модулем $f(x)$. І оберемо примітивний елемент $g(x) = 3x^3 + 3x^2 + 6$.

Далі обчислимо дискретні логарифми:

$a_0 = \log_{g(x)}(x) = 1028$;

$a_1 = \log_{g(x)}(x+1) = 1935;$

$a_2 = \log_{g(x)}(x+2) = 2054;$

$a_3 = \log_{g(x)}(x+3) = 1008;$

$a_4 = \log_{g(x)}(x+4) = 379;$

$a_5 = \log_{g(x)}(x+5) = 1780;$

$a_6 = \log_{g(x)}(x+6) = 223.$

Оберемо ціле число $d = 1702$ і обчислимо значення c_i за формулою $c_i = (a_i + 1702) \pmod{2400}$:

$c_0 = (1028 + 1702) \pmod{2400} = 2730 \pmod{2400} = 330;$

$c_1 = (1935 + 1702) \pmod{2400} = 3637 \pmod{2400} = 1237;$

$c_2 = (2054 + 1702) \pmod{2400} = 3756 \pmod{2400} = 1356;$

$c_3 = (1008 + 1702) \pmod{2400} = 2710 \pmod{2400} = 310;$

$c_4 = (379 + 1702) \pmod{2400} = 2081;$

$c_5 = (1780 + 1702) \pmod{2400} = 3482 \pmod{2400} = 1082;$

$c_6 = (223 + 1702) \pmod{2400} = 1925.$

Отже, $(\{330, 1237, 1356, 310, 2081, 1082, 1925\}, 7, 4)$ — публічний ключ, а $\left(x^4 + 3x^3 + 5x^2 + 6x + 2, 3x^3 + 3x^2 + 6, 1702\right)$ — приватний ключ.

Процес шифрування

Для цього процесу необхідно знати публічний ключ $(\{c_0, \ldots, c_{p-1}\}, p, h)$

1) Представити повідомлення m у вигляді двійкових рядків довжини $\lg \binom{p}{h}$, де $\binom{p}{h}$ — біноміальний коефіцієнт.
2) Розглянути m як двійкове представлення цілого числа. Перетворити це ціле число у двійковий вектор $M = (M_0, M_1, \ldots, M_{p-1})$ довжини p.
3) Обчислити $c = \sum_{i=0}^{p-1} M_i c_i \pmod{p^h - 1}$. c — зашифроване повідомлення.

Приклад 2.5. Зашифрувати повідомлення $m = 22$, використовуючи публічний ключ, отриманий в прикладі 2.4.

Оскільки, $\lg \binom{7}{4} = 5$, то спочатку представимо m у вигляді двійкого

рядка довжиною 5: $m = 10110$. Трансформуємо m в двійковий вектор $M = (1, 0, 1, 1, 0, 0, 1)$ довжини 7.

Обчислимо c

$c = (1 \cdot c_0 + 0 \cdot c_1 + 1 \cdot c_2 + 1 \cdot c_3 + 0 \cdot c_4 + 0 \cdot c_5 + 1 \cdot c_6) \pmod{2400} = (330 + 1356 + 310 + 1925) \pmod{2400} = 3921 \pmod{2400} = 1521.$

Отже, зашифрований текст $c = 1521$.

Процес дешифрування Щоб розшифрувати повідомлення, необхідно зробити такі кроки

1) Обчислити $r = (c - hd) \pmod{p^h - 1}$.
2) Обчислити $u(x) = g(x)^r \pmod{f(x)}$
3) Обчислити $s(x) = u(x) + f(x)$ многочлен степеня h над $\mathbb{Z}_p$.
4) Розкласти $s(x)$ на лінійні множники над $\mathbb{Z}_p$: $s(x) = \prod_{j=1}^{h}(x + t_j)$, де $t_j \in \mathbb{Z}_p$.
5) Обчислити двійковий вектор $M = (M_0, M_1, \ldots, M_{p-1})$. Компоненти, які є одиницями, мають індекси t_j, $1 \le j \le h$. Інші компоненти — нулі.
6) Вихідне повідомлення m отримується з M таким чином
   а) $m = 0$, $l = h$.
   б) Для кожного i $(1 \le i \le p)$: Якщо $M_{i-1} = 1$, тоді встановлюємо нові значення для $m = m + \binom{p-i}{l}$ і $l = l - 1$.

Теорема 2.1. *Довести, що алгоритм дешифрування дійсно дозволяє відновити оригінальний відкритий текст (повідомлення).*

Доведення. Зауважимо, що

$$\begin{aligned} u(x) &= g(x)^r \pmod{f(x)} \\ &\equiv g(x)^{c-hd} \equiv g(x)^{(\sum_{i=0}^{p-1} M_i c_i) - hd} \pmod{f(x)} \\ &\equiv g(x)^{\sum_{i=0}^{p-1} M_i (a_i + d)) - hd} \equiv g(x)^{\sum_{i=0}^{p-1} M_i a_i} \pmod{f(x)} \\ &\equiv \prod_{i=0}^{p-1} (g(x)^{a_i})^{M_i} \equiv \prod_{i=0}^{p-1} x^{M_i} \pmod{f(x)}. \end{aligned}$$

Отже, h коренів $s(x)$ всі лежать в $\mathbb{Z}_p$. І вони дають координати M, які є одиницями. □

Приклад 2.6. Розшифрувати повідомлення $c = 1521$, використовуючи ключ, отриманий в прикладі 2.4.

Спочатку обчислимо r:

$$r = (1521 - 4 \cdot 1702) \pmod{2400} = -5287 \pmod{2400} = 1913.$$

Тепер обчислимо $u(x)$ і $s(x)$:

$$u(x) = (3x^3 + 3x^2 + 6)^{1913} \pmod{x^4 + 3x^2 + 5x^2 + 6x + 2} = x^3 + 3x^2 + 2x + 5.$$
$$s(x) = x^3 + 3x^2 + 2x + 5 + x^4 + 3x^2 + 5x^2 + 6x + 2 = x^4 + 4x^3 + x^2 + x.$$

Далі необхідно розкласти $s(x)$ на лінійні множники:

$$s(x) = x(x + 2)(x + 3)(x + 6),$$

таким чином, $t_1 = 0$, $t_2 = 2$, $t_3 = 3$, $t_4 = 6$.

Отже, $M = (1, 0, 1, 1, 0, 0, 1)$. Трансформуємо це у текст і отримуємо, що $m = 22$.

Наведемо переваги цієї модифікації над стандартною рюкзачною криптосистемою.

1) Якщо параметри системи ретельно підібрані, то не існує жодної атаки на схему шифрування Шора—Рівеста, яку можна вдало здійснити. Зокрема, щільність набору рюкзака $(c_0, c_1, \ldots, c_{p-1})$ — це $\dfrac{p}{\lg(\max c_i)}$, що є достатньо великим значенням, а тому запобігає атакам низької щільності на цю криптосистему.
2) У прикладі наведено алгоритм лише для випадку, коли p — просте. Але насправді алгоритм поширюється і на випадок, коли базове поле $\mathbb{Z}_p$ заміюнється полем простого степеневого порядку.
3) Шифрування відбувається швидко. А для дешифрування потрібно значно більше часу.

Щодо недоліків цієї модифікації, то основним є те, що система буде незахищеною, якщо відома частина приватного ключа. І ще однією слабкістю є велика довжина публічного ключа.

РОЗДІЛ 3

ПРАКТИЧНА РЕАЛІЗАЦІЯ РЮКЗАЧНОЇ КРИПТОСИСТЕМИ

Основна задача третього розділу — написати програму за допомогою мови програмування Python для практичної реалізації стандартної рюкзачної криптосистеми Меркле—Хеллмана. Код програми наведено у Додатку А.

Спочатку користувачу необхідно ввести повідомлення у двійковому форматі. Програма рахує довжину цього повідомлення і записує у змінну k. Це потрібно для того, щоб довжина вихідного повідомлення і ключів була однакова.

Далі необхідно обрати суперзростаючу послідовність. З метою зменшення шансів злому, визначена фукнція *generate_sequence*, виклик якої щоразу генерує випадкову суперзростаючу послідовність довжиною k. Оскільки ціле число n має бути більше за суму всіх членів послідовності, то воно також генерується випадково таким чином: до загальної суми додається випадкове число з проміжку $(10; 50)$. Після цього користувач бачить віконце, в яке він самостійно має ввести значення числа u з умовою, що воно має бути взаємнопростим зі згенерованим числом n. Це обов'язкова умова, інакше — програма видасть помилку. Генерація приватного ключа завершена.

Перейдемо до створення публічного ключа. Для цього визначена функція *generate_public_key*, яка враховує приватний ключ, числа u і n, і обчислює значення q_i за формулою $q_i = u \cdot s_i \pmod{n}$. Отримана послідовність і буде публічним ключем.

Тепер перейдемо безпосередньо до шифрування вхідного повідомлення. Для цього визначена функція *encrypt*, яка працює таким чином: проходиться по всіх елементах повідомлення, обирає ті з них, які є одиницями, і додає елементи публічного ключа, які розташовані на відповідних місцях. Зашифрований текст зберігається у змінній *encrypted_text*.

Для процесу розшифрування спочатку потрібно обчислити u^{-1} і $l_i =$

$encrypted\_text \cdot u^{-1} \pmod{n}$. Значення зберігаються у змінних $u\_inverse$ і l_i відповідно. Щоб розшифрувати зашифрований текст, необхідно викликати функцію *decrypt*. Принцип її роботи такий:

1) Створюється порожній рядок *decrypted_text*, який згодом буде розшифрованим текстом.
2) Цикл проходиться по всіх елементах перевернутої суперзростаючої послідовності.
3) Якщо значення l_i більше або дорівнює найбільшому елементу послідовності, тоді до створеного рядка додаємо 1, а l_i зменшуємо на величину цього елемента.
4) Інакше — до рядка додаємо 0, а значення l_i залишається без змін.

Отриманий рядок *decrypted_text* і буде розшифрованим повідомленням.

На рисунку 3.1 зображено результат роботи програми.

```
Введіть повідомлення:101001
Приватний ключ: [45, 55, 106, 214, 428, 850]
n= 1723
Введіть число u:111
Публічний ключ: [1549, 936, 1428, 1355, 987, 1308]
Зашифрований текст: 4285
u^{-1}= 683
l_i= 1001
Розшифрований текст: 101001
```

Рис. 3.1. Результат роботи програми

Видно, що розшифрований текст і вхідне повідомлення ідентичні, значить, програма правильно виконала обчислення. Але для більшої впевненості проведемо обчислення для тих самих даних самостійно.

Приклад 3.1. Дано вхідне повідомлення $m = 101001$, $S_n = \{45, 55, 106, 214, 428, 85$ $n = 1723$, $u = 111$. Необхідно зашифрувати вхідне повідомлення, а потім розшифрувати його.

Спочатку обчислимо q_i за формулою $q_i = 111 \cdot s_i \pmod{1723}$:

$q_1 = 111 \cdot 45 \pmod{1723} = 1549;$

$q_2 = 111 \cdot 55 \pmod{1723} = 936;$

$q_3 = 111 \cdot 106 \pmod{1723} = 1428;$

$q_4 = 111 \cdot 214 \pmod{1723} = 1355;$

$q_5 = 111 \cdot 428 \pmod{1723} = 987;$

$q_5 = 111 \cdot 850 \pmod{1723} = 1308.$

Отже, послідовність $Q = [1549, 936, 1428, 1355, 987, 1308]$ — публічний ключ.

Перейдемо до шифрування:

$en = 1 \cdot 1549 + 0 \cdot 936 + 1 \cdot 1428 + 0 \cdot 1355 + 0 \cdot 987 + 1 \cdot 1308 = 4285$ — зашифрований текст.

Перейдемо до процесу дешифрування. Спочатку знайдемо u^{-1} таке що, $111 \cdot u^{-1} = 1 \pmod{1723}$. Тоді $u^{-1} = 683$. Обчислимо значення l_i:

$l_i = en \cdot u^{-1} \pmod{1175} = 4285 \cdot 683 \pmod{1723} = 1001.$

$1001 - 850 = 67 - 67 = 0$. Отже, розшифрований текст буде 10010.

Усі обчислені значення у прикладі 3.1 збіглись зі значеннями, які обчислила програма. Можна зробити висновок, що програма дійсно правильно шифрує і дешифровує повідомлення.

ВИСНОВКИ

Рюкзачна криптосистема Меркла—Хеллмана має велике історичне значення, адже вона була першою конкретною реалізацією криптосистем з відкритим ключем. З часом було виявлено, що ця криптосистема не є криптостійкою. Проте цей алгоритм часто використовують як ілюстративний: на його прикладі розглядають основні поняття та інструменти асиметричної криптографії.

Для усунення вразливостей і недоліків цієї криптосистеми було запропоновано багато модифікацій, які мали б підвищити рівень безпеки. Але їх потрібно регулярно оновлювати по мірі виявлення нових методів атак.

У кваліфікаційній роботі були розглянуті алгоритми стандартної криптосистеми, виявлені її переваги та недоліки, зокрема було на прикладі застосовано алгоритм LLL для злому. Також було досліджено декілька модифікацій. В процесі аналізу було виявлено, що рюкзачна криптоситема із застосуванням символу Лагранжа є значно безпечнішою і ефективнішою. Адже вона не лише збільшує час розшифрування, але й є стійкою до атак Шаміра, відкритого тексту і алгоритму LLL. Результатом практичної частини є програма, яка є практичною реалізацією криптосистеми Меркла—Хеллмана. Усі поставлені завдання було виконано.

СПИСОК ЛІТЕРАТУРИ

1. Arto Salomaa, Public-Key Cryptography // Springer. — 1996. — Vol. 2. — P. 55–124.
2. Bruce Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C // Wiley. — 1996. — P. 461–466.
3. Desmedt, Y.G., Skwirzynski, J.K. ed, What happened to the knapsack cryptographic scheme? // Netherlands: Kluwer Academic Publishers. — 1988. — Vol. 142. — P. 113–134.
4. Diffie W., Hellman M., New directions in cryptography // IEEE Transactions on information theory. — 1976. — Vol. IT–22. — P. 644–654.
5. Hamza B. Habib, Wadhah A. Hussein, Diana S. Mahdi, Improving the security of the Knapsack Cryptosystem by using Legendre Symbol // Turkish Journal of Computer and Mathematics Education. — 2021. — Vol. 12, No 11. — P. 2249 –2255.
6. Henk C. A. van Tilborg, Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial // Springer. — 1999. — Vol. 2. — P. 262–285.
7. Mark Stamp, Information security: principles and practice // John Wiley and Sons, Inc. — 2006. — P.128–134.
8. Menezes A., P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography // CRC Press. — 1996. — P. 300–306.
9. Merkle R. and Hellman M., Hiding information and signatures in trapdoor knapsacks // IEEE Transactions on Information Theory. — 1978. — Vol. IT-24, No. 5. — P. 525—530.
10. Ming Kin Lai, Knapsack Cryptosystems: The Past and the Future // University of California. — 2001.
11. Shamir A., A polynomial-time algorithm for breaking the basic Merkle–Hellman cryptosystem // IEEE Transactions on Information Theory. — 1984. — Vol. IT–30, No. 5. — P. 699–704
12. Марчук К. В. Рюкзачна криптосистема та її модифікації // Матеріали міжнародної наукової конференції, присвяченої 55-річчю факультету

математики та інформатики. — Чернівці. — 2023. — 262 с.

ДОДАТОК А

```
import random

#Функція для генерації випадкової суперзростаючої послідовності
def generates_sequence(k):
    sequence = [random.randint(1, 100)]
    while len(sequence) < k:
        next_element = sum(sequence) + random.randint(1, 10)
        sequence.append(next_element)
    return sequence

#Функція для генерації публічного ключа з приватного
def generate_public_key(private_key, u, n):
    public_key = [(u * element) % n for element in private_key]
    return public_key

#Функція для шифрування повідомлення
def encrypt(text, public_key):
    encrypted_text = sum(public_key[i] for i in range(len(text))
    if text[i] == '1')
    return encrypted_text

#Функція для розшифровки повідомлення
def decrypt (encrypted_text, private_key, l_i):
    decrypted_text = ''
    for element in list(reversed(private_key)):
        if l_i >= element:
            decrypted_text = '1' + decrypted_text
            l_i= l_i-element
        else:
            decrypted_text = '0' + decrypted_text
    return decrypted_text
```

```
text=input('Введіть повідомлення:') #Вхідний текст
k=len(str(text)) #Довжина вихідного тексту
private_key=generates_sequence(k) #Приватний ключ
print('Приватний ключ:', private_key)
n=sum(private_key)+random.randint(10,50) #Генерація числа n
print('n=', n)
u=int(input('Введіть число u:'))
public_key=generate_public_key(private_key, u, n) #Публічний ключ
print('Публічний ключ:', public_key)
encrypted_text=encrypt(text, public_key) #Зашифрований текст
print('Зашифрований текст:', encrypted_text)
u_inverse = pow(u, -1, n) #Обернене до u
print('u^{-1}=', u_inverse)
l_i=(encrypted_text * u_inverse) % n #Обчислення l_i
print('l_i=', l_i)
decrypted_text=decrypt(encrypted_text, private_key, l_i)
print('Розшифрований текст:', decrypted_text)
```