НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Факультет прикладної математики

Кафедра прикладної математики

«До захисту допущено»

Завідувач кафедри

_____ О. Р. Чертов

«___» _____ 2016 р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.040301 «Прикладна математика»

на тему: Застосування LLL-методу для криптоаналізу асиметричної системи шифрування Меркла-Хеллмана

Виконав: студент IV курсу, групи КМ-22

Бучинський Олександр Юрійович _____

Керівник Старший викладач Копичко С.М. _____

Консультанти:

- із особливих питань Д.т.н., професор Савчук М.М. _____

- із нормоконтролю Старший викладач Мальчиков В.В. _____

Рецензент Доцент, к.т.н., доцент Плахотний М.В. _____

Засвідчую, що в цій дипломній роботі немає запозичень із праць інших авторів без відповідних посилань.
Студент_____

Київ — 2016

Національний технічний університет України

«Київський політехнічний інститут»

Факультет прикладної математики

Кафедра прикладної математики

Рівень вищої освіти — перший (бакалаврський)

Напрям підготовки 6.040301 «Прикладна математика»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ О. Р. Чертов

«___»_____2016р

.

ЗАВДАННЯ

на дипломну роботу студенту

Бучинському Олександру Юрійовичу

1. Тема роботи: «Застосування LLL-методу для криптоаналізу асиметричної системи шифрування Меркля-Хеллмана»,

керівник роботи Копичко Сергій Миколайович, старший викладач ,

затверджені наказом по університету від «06» травня 2016 р. № 1499-С.

2. Термін подання студентом роботи: «9» червня 2016 р.

3. Вихідні дані до роботи: система, яка шифрує та розшифровує дані, та модуль, який реалізує криптоаналіз системи LLL-алгоритмом.

4. Зміст роботи: ознайомитись з принципами побудови системи шифрування Меркля-Хеллмана, реалізувати відповідне програмне забезпечення, дослідити шляхи вдалого криптоаналізу реалізованої системи за допомогою LLL-алгоритмом.

5. Перелік ілюстративного матеріалу:знімки екранних форм.

6. Дата видачі завдання: «22» лютого 2016 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури за тематикою та збір даних	12.11.2015	
2	Проведення порівняльного аналізу математичних методів	14.12.2015	
3	Проведення порівняльного аналізу програмних рішень	24.12.2015	
4	Підготовка матеріалів першого розділу роботи	01.02.2016	
5	Розроблення математичного забезпечення системи	01.03.2016	
6	Підготовка матеріалів другого розділу роботи	15.03.2016	
7	Підготовка матеріалів третього розділу роботи	05.04.2016	
8	Розроблення програмного забезпечення системи	15.04.2016	
9	Підготовка матеріалів четвертого розділу роботи	03.05.2016	
10	Оформлення пояснювальної записки	01.06.2016	

Студент _____ Бучинський О. Ю.

Керівник роботи _____ Копичко С. М.

АНОТАЦІЯ

Дипломну роботу виконано на 46 аркушах, вона містить 2 додатки та перелік посилань на використані джерела з 6 найменувань. У роботі наведено 8 рисунків та 1 таблиця.

Метою даної дипломної роботи є створення математичного та програмного забезпечення для асиметричної системи шифрування Меркеля-Хеллмана, та дослідження застосування LLL- алгоритму для її вдалого криптоаналізу.

У роботі було розглянуто побудову системи шифрування, що базується на NP-повній задачі про укладку ранця. Розглянуто шляхи криптоаналізу побудованої системи за допомогою LLL-алгоритму пошуку короткого базису цілочисельної решітки.

Розроблено автоматизовану систему для шифрування та розшифрування даних, також програмно реалізований її криптоаналіз. Проведення тестування розробленої системи.

Ключові слова: асиметричне шифрування,задача про укладку ранця, модульне множення, LLL-алгоритм пошуку короткого базису цілочисельної решітки.

ABSTRACT

The thesis is presented in 46 pages. It contains 2 appendixes and bibliography of 6 references. Eight figures and one table are given in the thesis.

The goal of the thesis is to develop mathematical and software tools for asymmetric system of enciphering Merkela-Hellmana and research of using LLL-algorithm for its cryptoanalysis.

System of enciphering which was built on a NP-full task of laying of a backpack is considered. The way of cryptoanalysis for this system with LLL- algorithm for build short basis integral lattices is discussed.

The automated system for enciphering and enciphering are developed, also cryptoanalysis for its is developed too. The developed system is tested.

Keywords: asymmetric system of enciphering, task of laying of a backpack, modular multiplication, LLL- algorithm for build short basis integral lattices.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DFD - Data flow diagram.

LLL - Lenstra A.K., Lenstra H.W., Lovasz L.

NP-клас - клас задач, що вирішуваною за поліноміальний час виявляється перевірка: буде вдало вгадане рішення проблеми вірним або ні.

NP-повна задача - в теорії алгоритмів та теорії складності це задача, що належить до класу NP та всі задачі з класу NP можна звести до неї за поліноміальний час.

ВСТУП

Масове упровадження комп'ютерів в усі сфери діяльності людини сприяло збільшенню обсягу інформації збереженої в електронному вигляді. Тому особливої актуальності набули і засоби захисту та приховування інформації, один з який - шифрування. Головним принципом якого є унеможливлення прочитати інформацію без спеціального ключа.

В залежності від ключа виділяють симетричні, та асиметричні методи шифрування. У симетричних методах один ключ, який зберігається в секреті, служить і для шифрування, і для дешифрування. Асиметричні алгоритми, або метод відкритого ключа, передбачає застосування в парі двох відмінних ключів, а саме секретний та відкритий. Саме системі з асиметричним методом шифрування Меркла-Хеллмана присвячений дана дипломна робота.

Актуальність асиметричних систем полягає у відсутності необхідності передачі ключів, що є головною проблемою симетричних систем. Також використовують змішані системи, де ключ симетричної системи передається повільнішим асиметричним алгоритмом, а далі повідомлення шифрується за допомогою біль швидкого симетричного алгоритму.

Система Меркла-Хеллмана має перевагу у швидкодій над багатьма популярними асиметричними системами, проте вона не завжди є надійною. Саме через швидкодію алгоритму актуальним є дослідження шляхів її криптоаналізу для подальшого їх використання у побудові більш стійкою системи, що базується на даному алгоритмі.

1. ПОСТАНОВКА ЗАДАЧІ

Метою даної дипломної роботи є створення математичного та програмного забезпечення для асиметричної системи шифрування Меркеля-Хеллмана, та LLL-алгоритму для її криптоаналізу. Використовуючи розроблені засоби дослідити залежність вдалого криптоаналізу алгоритму від вхідних даних системи.

Для виконання поставленої задачі необхідно виконати наступні завдання:

a) Ознайомитись з теоретичними матеріалами стосовно системи шифрування побудованої на задачі про укладку ранця.

b) Проектування програми, що буде мати можливість шифрування та розшифрування даних на основі введеного ранцевого вектору за допомогою асиметричного алгоритму.

c) Ознайомлення з LLL-алгоритмом побудови зменшеного базису цілочисельної решітки, та його застосування для криптоаналізу побудованої системи.

d) Програмна реалізація LLL-алгоритму.

e) За допомогою розроблених програмних продуктів провести дослідження шляхів вдалого криптоаналізу системи, та зробити відповідні висновки, що до ефективності криптоаналізу .

2. АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ПРОЕКТУВАННЯ СИСТЕМИ ТА ЇЇ КРИПТОАНАЛІЗУ

2.1 Шифрування, методи шифрування

Шифрування – це оборотне перетворення даних, з метою приховування інформації. Шифрування відбувається із застосуванням криптографічного ключа. Ключ – це певна кількість символів, сформованих вільним чином з символів, що доступні у системі шифрування.

Припускають, що шифрування з'явилося приблизно 4 тис. років тому. Першою відомою пам'яткою шифрування прийнято вважати єгипетський текст, який було створено напевно десь в 1900 році до нашої ери., у якому використовувались другі символи замість відомих єгипетських ієрогліфів. Загалом виділяють два методи шифрування симетричне та асиметричне.

У симетричному шифруванні один ключ, який зберігається в секреті, служить і для шифрування, і для дешифрування. Симетричні алгоритми шифрування можна класифікувати на потокові та блочні. Потокові алгоритми шифрування поетапно опрацьовують текстове повідомлення. Блочні алгоритми співпрацюють з блоками зафіксованого розміру. найчастіше розмір блоку дорівнює 64 бітам.

Симетричні алгоритми шифрування також можуть використовуватися не самостійно. У новітніх крипто системах, застосовуються комбінації симетричних та асиметричних алгоритмів, з метою отримання переваг обох схем.

У симетричному шифруванні можна виділити деякі переваги, такі як велика пропускна здатність, завдяки спеціальному проектуванню; ключі мають невеликий розмір; дані шифри можна застосовувати як основу для будування різноманітних крипто графічних механізмів, включаючи з випадковими генераторами чисел, обчислювально-ефективними схемами розпису та тому подібне.

Серед недоліків даного шифрування слід відзначити те, що у кожній немаленькій мережі необхідно використовувати значну кількість ключів; при зв'язку між декількома особами необхідно досить часто змінювати ключі; коли існує зв'язок між двома особами ключ слід засекречувати на двох кінцях.

Асиметричне шифрування, або метод відкритого ключа, передбачає застосування в парі двох відмінних ключів, а саме секретний та відкритий. Відповідно до назви відкритий ключ безперешкодно розміщується у мережі, логічно, що секретний ключ весь час тримається в таємниці.

Принцип роботи асиметричного шифрування можна простежити на прикладі роботи кейсу, для якого застосовують два ключа, першим кейс зачиняють, а другим - відчиняють.

Через певні вади швидкості дії асиметричного методу, його доводиться застосовувати разом із симетричним, так як він працює на декілька порядків швидше.

Асиметричне шифрування допомагає одержувачеві контролювати цілісність інформації, яка передається. [1].

2.2 Поняття асиметричної криптосистеми

Відправною точкою розвитку асиметричної криптографії стала робота «Нові напрямки в сучасній криптографії» Уітфілда Діффі та Мартіна Хеллмана, опублікована у 1976 році. Автори цієї роботи запропонували першу з відомих схем розповсюдження ключів по відкритих каналах.

В асиметричному шифруванні відкритий та секретний ключі співпрацюють у парі, тобто коли інформація шифрується відкритим ключем, то розшифровування відбувається тільки відповідним секретним ключем та навпаки. Неможливим є використовування відкритого ключа із однієї пари та

секретного ключа із іншої пари. Всі пари асиметричних ключів пов'язані за допомогою математичних залежностей.

Ідея криптографії з відкритим ключем дуже тісно пов'язана з ідеєю односторонніх функцій, тобто таких функцій f(x), що знаючи x дуже просто знайти f(x), проте визначення x за відомого значення f(x) обчислювально неможливе. Саму односторонню функцію застосовувати недоцільно, оскільки з її використанням можна зашифрувати повідомлення, проте розшифрувати – ні. Тому в асиметричній криптографії використовуються односторонні функції з лазівкою – секретом, за допомогою якої легальний користувач може отримати розшифровану інформацію.

Основні кроки побудови криптосистем з відкритим ключем:

1. В основі криптосистеми лежить задача P, яка має бути важковирішуваною в сенсі теорії складності, тобто не повинно існувати алгоритму, який вирішує усі варіанти завдання P за поліноміальний час відносно розміру завдання. Прийнятніше, щоб не лише складність у гіршому випадку, а й складність в середньому випадку для задачі P була досить високою.

2. Виділяється легка підзадача P_{easy} з P. P_{easy} повинна вирішуватися за поліноміальний час, прийнятніше навіть за лінійний час.

3. Перетасувати або збовтати P_{easy} так, щоб результуюче завдання $P_{shuffle}$ не мало ніякої подібності з P_{easy}. Задача $P_{shuffle}$ повинна, принаймні, виглядати як оригінальне важковирішуване завдання P .

4. Відкривається $P_{shuffle}$ з описом, як вона може бути використана в якості ключа шифрування. Інформація про те, як з $P_{shuffle}$ отримати P_{easy}, тримається в секреті як секретна лазівка .

5. Деталі криптосистеми конструюються у такий спосіб, щоб алгоритм розшифрування був істотно різним для криптоаналітика і легального одержувача. Поки перший вирішує $P_{shuffle}$ (маючи вигляд важковирішуваного завдання P), останній може використати секретну лазівку і вирішувати тільки P_{easy} .[2].

2.3 Задача про рюкзак

Ранцевим вектором $A = (a_1, \dots, a_n)$ називається впорядкований набір із $n, n \geq 3$, різних натуральних чисел a_i. Входом задачі про рюкзак називається пара (A, α), де A – рюкзачний вектор, а α – натуральне число. Розв'язком для входу (A, α) буде така підмножина із A, сума елементів якої рівна α.

В найбільш відомому варіанті задачі про рюкзак необхідно визначити, чи має даний вхід (A, α) розв'язок. В варіанті, що використовується в криптографії, потрібно для даного входу (A, α) побудувати розв'язок, знаючи, що такий розв'язок існує.

Рішення даної задачі шляхом перебору для достатньо великих рюкзаків знаходиться за межами обрахункових можливостей сучасних комп'ютерів, а алгоритмів, що працюють значно швидше повного перебору не існує. Натомість елементарним є отримання місткості рюкзаку, знаючи предмети, які йому належать тобто вона є NP-важкою задачею.

Оскільки є бажаною однозначність розшифрування, ранцеві вектори A повинні мати таку властивість, що для кожного α всі входи (A, α) мають не більше одного розв'язку. Такі рюкзачні вектори називаються ін'єктивними. Саме таким є надзростаючий ранцевим вектор.

Набір A $(a_1, a_2, \dots, a_n)$ називається таким, що надзростає, якщо кожне наступне число в ньому більше суми усіх попередніх, тобто:

$$a_n > \sum_1^{n-1} a_j, \text{ для } j = 2,3 \dots n \tag{2.1}$$

Ця задача належить підкласу легких задач укладання рюкзака. Немає необхідності в повному переборі при рішенні відповідночї задачі досить переглянути A один раз справо на ліво. Для заданого K (розміру рюкзака) ми спочатку перевіряємо $K > a_n$. якщо відповіддю буде ні, a_n не може входити в

шукану суму. Якщо ж відповіддю буде так , то a_n повинне входити в суму. Це витікає з того факту, що уся решта a_i- в сумі дає число, менше K. Рухаючись так по усьому вектору легко знайти розв'язок даної задачі.

Розглянемо ранцевий вектор A, ціле число $m > \max A$ і натуральне $t < m$ таке, що найбільший спільний дільник $(t, m) = 1$. Якщо $B = (b_1, \dots, b_n)$ такий вектор, що

$$b_i = (ta_i, \mathrm{mod}\ m), \text{для } i = 1, \dots, n, \tag{2.2}$$

то говорять, що вектор B отриманий із A за допомогою модульного множення відносно модуля m і множника t, тобто відносно пари (m, t). Умова $(t, m) = 1$ гарантує існування оберненого числа $t^{-1} = u$, такого що $tu \equiv 1 \bmod m$ і $1 \leq u < m$. Це означає, що також і навпаки A можна отримати із B модульним множенням відносно m і u . (Ясно, що $m > \max B$, оскільки кожне b_i береться по mod m.)

Якщо умова $m > \max A$ замінюється сильнішою умовою $m > \sum_{i=1}^{n} a_i$, то говорять, що B отримується із A сильним модульним множенням відносно m і t [2].

2.4 Система Меркла-Хеллмана

Розглянемо принцип побудови криптосистеми Меркла-Хеллмана. Засновник криптосистеми обирає A, t, m, B так, що вектор A є надзростаючим, а B отримується із A сильним модульним множенням відносно m і t. Вектор

B розкривається як ключ зашифрування і двійкові блоки довжини n відправляються до проектувальника як числа β, отримані за допомогою вектора B. Перехоплювач повідомлень повинен розв'язувати задачу про рюкзак для входу (B, β) . Засновник криптосистеми обчислює $\alpha = (u\beta, \mathrm{mod}\, m)$ і розв'язує задачу про рюкзак для входу (A, α) . Наступна лема доводить той факт, що така система працюватиме.

Лема 1 Припустимо, що $A = (a_1, \dots, a_n)$ надзростаючий вектор і вектор B отриманий із A сильним модульним множенням відносно m і t. Припустимо, що $u \equiv t^{-1} \mathrm{mod}\, m$, β – довільне натуральне число і $\alpha = (u\beta, \mathrm{mod}\, m)$. Тоді справедливі наступні твердження:

1) Задачу про рюкзак (A, α) можна розв'язати за лінійний час; якщо розв'язок існує, то він єдиний;

2) Задача про рюкзак (B, β) має не більше одного розв'язку;

3) Якщо існує розв'язок для входу (B, β), то він співпадає з єдиним розв'язком для входу (A, α) [2].

2.5 Підход А. Шаміра для знаходження елементів модульного множення

При побудові алгоритму не обов'язково шукати зворотний множник u та модуль м, які дійсно використовувалися творцем криптосистеми. Нас влаштує будь-яка пара (u, m) за умови, що u і m, задовольняють обмеженням, що накладаються на модульне множення щодо вектора B, що вектор A, що виникає в результаті такого модульного множення, є надзростаючим і що m перевершує суму компонент . (звідси випливає, що B виходить з A сильним модульним множенням)Також пари (u, m) будемо називати ними парами. Як тільки ми знайдемо хоча б одну секретну пару, ми зможемо застосувати лему 1 і почати розшифрування, використовуючи отриманий надзростаючий вектор. І це

абсолютно не залежить від того, чи будуть знайдена секретна пара і надзростаючий вектор тими, що реально використовувалися творцем криптосистеми. З іншого боку, існування принаймні однієї такої секретної пари гарантується тим, що творець криптосистеми одну таку пару використовував.

Для того щоб знайти секретну пару (u, m), розглянемо функції. $b_i u \pmod m$, для i=1,2..,n , графік якої складається з прямолінійних відрізків, а значення $i = \frac{pm}{b_i}$, p = 1, 2 ,.. є точками розриву.

$b_1 u = a_1 \pmod m$, де u - зворотній множник, який шукається. Оскільки a_1 є першою компонентою надзростаючого вектора і m перевищує суму всіх компонент, то a_1 має бути дуже мало в порівнянні з m. Звідси випливає, що значення u з секретної пари повинно бути близько до деякого мінімуму функції b_1.

Аналогічно розмірковуючи, приходимо до того, що значення u з секретної пари повинно бути близько до деякого мінімуму функції b_2. Це тягне (за нерівністю трикутника), що якісь два мінімуму функцій b_1 і b_2 повинні бути близькі один до одного.

Діючи таким же чином, можна розглянути і інші такі функції. Той факт, що значення u з секретної пари близько до деякого мінімуму кожної такої кривої, означає, що всі ці мінімуми близькі один до одного. Таким чином, замість того щоб намагатися знайти саме n, ми можемо намагатися знайти "точки накопичення" мінімумів кривих. Це рівносильно побудові деякого малого інтервалу, що містить мінімуми кожної з узятих пилкоподібних кривих. Виходячи з цього інтервалу, ми також знайдемо і значення u з секретної пари.

Цей алгоритм не завжди знаходить елементи модульного множення та важко дослідити залежність між ранцевим вектором та степенем поліному, що описую швидкості роботи алгоритму [2,3].

2.6 Застосування LLL-алгоритму

Розглянемо алгоритм знаходження розв'язку рюкзака, що базується на LLL-алгоритмі зменшення базису решітки. Цей алгоритм був запропонований у 1985 році Андрієм Одлижко та Джефрі Лагаріасом .

Вхід алгоритму. $A = (a_1, \dots, a_n)$ – рюкзачний вектор; $\alpha = \sum_{i=1}^{n} a_i x_i, x_i \in \{0, 1\}$.

Вихід алгоритму. У разі успішного завершення роботи алгоритму $x = (x_1, \dots, x_n)$ – двійковий вектор, що дає розв'язок для рюкзака.

Кроки алгоритму:

1. Взяти наступні вектори в якості базису $[b_1, \dots, b_{n+1}]$ для $(n + 1)$ –мірної цілочисельної решітки $L = L(A, \alpha)$:

$$b_1 = \left(1, 0, \dots, 0, \left[\frac{1}{2}\sqrt{n}\right] a_1\right)$$

$$b_2 = \left(0, 1, \dots, 0, \left[\frac{1}{2}\sqrt{n}\right] a_2\right)$$

$$\vdots$$

$$b_n = \left(0, 0, \dots, 1, \left[\frac{1}{2}\sqrt{n}\right] a_n\right)$$

$$b_{n+1} = \left(\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2}, \left[\frac{1}{2}\sqrt{n}\right] \alpha\right).$$

2. Знайти зменшений базис $[b_1^*, \dots, b_{n+1}^*]$ решітки L використовуючи LLL-алгоритм.

3. Для кожного вектора $b=(b_1, \dots, b_{n+1})$ зведеного базису , для якого $b_{n+1} = 0$ та $b_i \epsilon \left\{-\frac{1}{2}, \frac{1}{2}\right\}$ при i=1,2,..,n покласти:

3.1 $x_i = b_i + \frac{1}{2}$ для i=1,2,..,n

Якщо $\sum_{i=1}^{n} x_i a_i = \alpha$, тоді $(x_1, \dots, x_n)$ - розв'язок задачі.

3.2 $x_i = b_i - \frac{1}{2}$ для i=1,2,..,n

Якщо $\sum_{i=1}^{n} x_i a_i = \alpha$, тоді $(x_1, \dots, x_n)$ - розв'язок задачі.

В противному випадку алгоритм не призводить до знаходження розв'язку задачі. Оскільки задача про рюкзак є NP-повною задачею, цей алгоритм не завжди успішно завершуватиме роботу.

Введемо поняття густини рюкзака. Густиною $d(A)$ рюкзачного вектора $A = (a_1, \dots, a_n)$ називатимемо величину

$$d(A) = \frac{n}{\max(\log_2 a_i)} \tag{2.3}$$

У термінах ранцевих криптосистем з відкритим ключем $d(A)$ можна виразити як відношення:

$$d(A) \cong \frac{\text{кількість біт у відкритому тексті}}{\text{середня кількість біт у зашифрованому тексті}}. \tag{2.4}$$

Якщо густина ранцевого вектору не перевищує 0,9048, то описаний алгоритм з ймовірність близькою до 1 знаходить розв'язок задачі про укладку ранцю [4].

2.7 Висновки до розділу

Було розглянуто основні поняття шифрування, його методи, та основні принципи побудови системи шифрування з відкритим ключем. Визначено алгоритм побудови системи шифрування Маркла-Хеллмана основаної на задачі про укладку ранцю. Переглянуто два методи криптоаналізу даної системи, та обрано з поміж них LLL-алгоритм, бо він має більшу швидкодію, та може бути

вдало застосований для більш широкого класу ранцевих векторів у порівнянні з підходом Шаміра. Також, враховуючи невисокий коефіцієнт густини для надзростаючого ранцевого вектору, ймовірності вдалої роботи LLL-алгоритму повиннабути близькою до 1, в той час з використанням методу пошуку елементів модульного множення невідомою є ймовірність вдалого криптоаналізу .

3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Цілочисельна решітка та її базис. LLL-зменшений базис, та його властивості

Нехай вектора $b_1, \dots, b_n \in R^m$ лінійно незадежні та $n \leq m$ множина $L = L(b_1, \dots, b_n) = Zb_1 + Zb_2 + \dots + Zb_n$ всіх цілочисельних лінійних комбінацій векторів $b_1, \dots, b_n$ називається решітка розмірністю n. Множина $\{b_1, \dots, b_n\}$ називається базисом решітки.

Для означення LLL-зменшеного базису та його властивостей, а також важливою складовою LLL-алгоритму зменшення базису решітки є процедура ортогоналізації Грама-Шмідта.

Процедура ортогоналізації Грама-Шмідта виглядає наступним чином. Нехай $b_1, \dots, b_n$ – лінійно незалежні вектори в R^n. Вектори $b_1^*, \dots, b_n^*$ визначаються як:

$$b_1^* = b_1,\ b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^* \qquad (3.1)$$

де $\mu_{ij} = \dfrac{b_i, b_j^*}{b_j^*, b_j^*}$,, $i = 2, \dots, n,\ 1 \leq j < i.$

Тоді вектори $b_1^*, \dots, b_n^*$ - попарно ортогональні.

Алгоритм процедури ортогоналізації Грама-Шмідта, запропонований, виглядає наступним чином.

Вхід алгоритму. Лінійно незалежні вектори $b_1, \dots, b_n \in R^m, m \geq n$.

Вихід алгоритму. Ортогональна система векторів $b_1^*, \dots, b_n^*$.

Кроки алгоритму:

1 Покласти $b_1^* = b_1, B_1 = \|b_1^*\|^2$.

2 Для $i = 2, 3, \dots, n$ виконати наступні дії:

2.1 Покласти $b_i^* = b_i$.

2.2 Для $j = 1, \dots, i - 1$обчислити

$$\mu_{i,j} = \frac{(b_i, b_j^*)}{B_j}, b_i^* = b_i^* - \mu_{i,j} b_j^*.$$

2.3 Покласти $B_i = \|b_i^*\|^2$.

3 Результат $\{b_1^*, b_2^* \dots, b_n^*\}$.

Якщо точність обчислень складає l цифр, то складність алгоритму дорівнює $O(n^3 l^2)$.

Набір векторів $b_1, \dots, b_n$ називається LLL-зменшеним базисом решітки $L \subseteq R^n$, якщо

$$\|b_i{}^*\|^2 \geq (\frac{3}{4} - \mu_{i,i-1}{}^2)\|b_{i-1}{}^*\|^2, \ 1<i\leq n \qquad (3.2)$$

де $b_1{}^*, \dots, b_n{}^*, \mu_{ij}$ – отримані за допомогою процесу ортогоналізації Грама-Шмідта.

Розглянемо основні властивості LLL-зменшеного базису.

Якщо $\{b_1, b_2, \dots, b_n\}$ – LLL-зменшений базис решітки $L \subset R^m$, то для всіх $1 \leq j < i \leq n$ виконується нерівність (3.3).

$$\|b_j^*\|^2 \leq 2^{i-j}\|b_i^*\|^2 . \qquad (3.3)$$

Доведення. Із визначення LLL-зменшеного базису маємо:

$$\frac{3}{4}\|b_j^*\|^2 \leq \|b_{j+1}^*\|^2 + \mu_{j+1,j}^2\|b_j^*\|^2 \leq \|b_{j+1}^*\|^2 + \frac{1}{4}\|b_j^*\|^2,$$

тобто $\|b_j^*\|^2 \leq 2\|b_{j+1}^*\|^2$. Далі по індукції: із нерівності $\|b_j^*\|^2 \leq 2^{i-j}\|b_i^*\|^2$ отримуємо $2^{i+1-j}\|b_{i+1}^*\|^2 \geq 2^{i-j}\|b_i^*\|^2 \geq \|b_j^*\|^2$.

Зауваження. Для $j = 1$ отримуємо:

$$\|b_1^*\|^2 \leq 2^{i-1}\|b_i^*\|^2, 1 \leq i \leq n. \quad (3.4)$$

Теорема 1. Для LLL-зменшеного базису $\{b_1, b_2, ..., b_n\}$ решітки $L \subset R^m$ справедливі наступні твердження:

1) $\|b_i^*\|^2 \leq 2^{j-1}\|b_j^*\|^2$ для всіх $1 \leq i < j \leq n$.

2) $d(L) \leq \prod_{i=1}^{n}\|b_i\| \leq 2^{\frac{n(n-1)}{4}} d(L)$.

3) $\|b_1\|^2 \leq 2^{\frac{n-1}{4}} \sqrt[n]{d(L)}$.

Теорема 2. Для LLL-зменшеного базису $\{b_1, b_2, \dots, b_n\}$ решітки L і будь-якого ненульового вектора $x \in L$ $\|b_1\| \leq 2^{\frac{n-1}{2}}\|x\|$.

Теорема 3. LLL-зменшений базис $\{b_1, b_2, \dots, b_n\}$ решітки L є мінімальним в тому сенсі, що для будь-яких лінійно незалежних векторів $x_1, x_2, \dots, x_t$ решітки L для всіх $1 \leq i \leq t$ виконується нерівність

$$\|b_i\| \leq 2^{\frac{n-1}{2}} max_{1 \leq i \leq t}\|x_j\|. \text{ [4,5]}. \quad (3.5)$$

3.2 LLL-алгоритм зменшення базису решітки

Розглянемо алгоритм побудови зменшеного базису цілочисельної решітки, що був запропонований Арджeном Ленстре, Хендріком Ленстре та Лазло Ловасом у 1982 році .

На початку роботи алгоритму заданий $b_1, \dots, b_n$ – деякий базис решітки L. Після завершення роботи алгоритму $b_1, \dots, b_n$ – LLL-зменшений базис.

Проводиться індукція по $k \in \{1, 2, \dots, n+1\}$. Спочатку $k = 2$; коли k досягає значення $n + 1$, алгоритм завершує роботу і повертає LLL-зменшений базис.

Для кожного k символом $(*)_k$ ми будемо позначати сукупність нерівностей

$$\begin{cases} |\mu_{ij}| \le \frac{1}{2}, \quad 1 \le j \le i \le k, \\ |b_i^* + \mu_{ii-1} b_{i-1}^*|^2 \ge \frac{3}{4}|b_{i-1}^*|^2, 1 < i < k. \end{cases} \qquad (*)_k$$

Якщо $k = 2$, то умови $(*)_2$ виконані, оскільки для i вийде пуста множина значень $1 < i < 2$. Якщо $k = n + 2$, то $(*)_{n+1}$ означає, що базис зведений (за визначенням).

Припустимо, що для деякого $k, 1 < k < n + 1$, виконуються нерівності $(*)_k$. Нам потрібно забезпечити виконання $(*)_{k+1}$. Для початку забезпечимо виконання нерівності

$$|\mu_{k,k-1}| \le \frac{1}{2}. \qquad (3.6)$$

Якщо (3.6) виконується, то рухаємося далі. Якщо ні, знаходимо r – найближче ціле до $\mu_{k,k-1}$ і замінюємо b_k на

$$b_k - rb_{k-1} = b_k^* + \sum_{j=1}^{k-2} (\mu_{kj} - r\mu_{k-1,j}) b_j^* + (\mu_{k,k-1} - r) b_{k-1}^*.$$

Тоді коефіцієнт $\mu_{k,k-1}$ заміниться на $\mu_{k,k-1} - r$. Коефіцієнти $\mu_{k,j}$ заміняться на $\mu_{k,j} - r\mu_{k-1,j}$, $j = 1, \dots, k - 2$. Решта коефіцієнтів $\mu_{i,j}$ і вектори b_i^* при $i < k$ та при $i > k$, а також вектори b_k^* не зміняться. Дійсно, b_i^* є проекцією b_i на $L_{об}(b_1, \dots, b_{i-1})^{\perp}$; після заміни b_k на $b_k - rb_{k-1}$ дані лінійні оболонки не змінюються, а отже не змінюються всі b_i^*. Далі, оскільки $\mu_{ij} = (b_i, b_i^*)/(b_j^*, b_j^*)$, то при $i \ne k$, μ_{ij} не змінюються через те, що b_i та b_j^* залишилися незмінними.

Продовжимо забезпечення умов $(*)_{k+1}$, вважаючи, що (3.6) виконується.

Перший випадок. Нехай $k \geq 2$ і виконується нерівність

$$\left|b_k^* + \mu_{k,k-1} b_{k-1}^*\right|^2 < \frac{3}{4}|b_{k-1}^*|^2 \tag{3.7}$$

Тоді ми міняємо місцями вектори b_k та b_{k-1}. При цьому змінюються вектори b_{k-1}^* і b_k^* та коефіцієнти $\mu_{k,k-1}, \mu_{k-1,j}, \mu_{k,j}, \mu_{i,k-1}, \mu_{i,k}$, де $j < k-1$ або $i > k$. Решта b_i, b_i^* і $\mu_{i,j}$ не зміняться з тих же причин, що й раніше, тобто тому, що $L_{\text{об}}(b_1, \dots, b_{i-1})^\perp$ при $i \neq k, k-1$ залишається тією ж, і вектори b_i^* при $i \neq k, k-1$ не зміняться.

При заміні місцями b_k та b_{k-1} вектор $b_k^* + \mu_{k,k-1} b_{k-1}^*$, який раніше дорівнював проекції b_k на $L_{\text{об}}(b_1, \dots, b_{k-1})^\perp$, а тепер він дорівнює проекції нового вектора b_{k-1} на $L_{\text{об}}(b_1, \dots, b_{k-2})^\perp$. Далі b_{k-1}^* дорівнював проекції b_{k-1} на $L_{\text{об}}(b_1, \dots, b_{k-2})^\perp$, а тепер це і є проекція нового вектора b_k на $L_{\text{об}}(b_1, \dots, b_i)^\perp$. З нерівності (3.7) слідує, що при заміні значення $\|b_{k-1}^*\|^2$ зменшилося більше, ніж у ¾ раза.

Здійснивши цю заміну місцями b_k і b_{k-1}, ми замінюємо k на $k-1$ і опиняємося в умовах $(*)_{k-1}$.

Другий випадок. Нехай або $k = 1$, або

$$\left|b_k^* + \mu_{k,k-1} b_{k-1}^*\right|^2 \geq \frac{3}{4}|b_{k-1}^*|^2. \tag{3.8}$$

Якщо $k = 1$, то присвоюємо k значення 2 і продовжуємо процес, тобто забезпечуємо виконання $(*)_2$.

Якщо виконана умова (3.8) і $k > 1$, то забезпечуємо виконання нерівностей

$$\left|\mu_{kj}\right| \leq \frac{1}{2}, 1 \leq j \leq k-1 \tag{3.9}$$

(для $j = k-1$ (3.9) уже виконується в силу (3.7)).

Нехай l – найбільший номер, для якого $|\mu_{kl}| > \frac{1}{2}$. Тоді $l \leq k - 2$. Візьмемо r – найближче ціле до μ_{kl} – і замінимо b_k на $b_k - rb_l$. При цьому μ_{kj} замінюється на $\mu_{kl} - r$. Решта μ_{ij} і всі вектори b_i^* залишаться при цьому незмінними. Ми продовжуємо цей процес, зменшуючи l, поки не досягнемо значення $l = 1$. В цьому випадку ми забезпечуємо виконання умов $(*)_{k+1}$.

Якщо ми досягли виконання умов $(*)_{n+1}$, то алгоритм зупиняється, оскільки $b_1, \dots, b_n$ утворюють зменшений базис; в протилежному випадку ми продовжуємо процес, тобто збільшуємо значення k.

Доведемо, що алгоритм закінчує роботу. Позначимо через

$$d_i = det\left\|(b_j, b_l)\right\|_{1 \leq j,l \leq i}, \ i = 1, \dots, n. \tag{3.10}$$

Очевидно, що

$$d_i = det(\|b_j\| \cdot \|b_l\|^T) = \left(det\|b_j\|_{1 \leq j \leq i}\right)^2 = \left(det\|b_j^*\|_{1 \leq j \leq i}\right)^2 =$$

$$= det(\|b_j^*\| \cdot \|b_l^*\|^T) = \prod_{j=1}^{i} |b_j^*|^2.$$

Оскільки 2 випадок спрацьовує за скінченну кількість операцій і збільшує значення k на одиницю, то нам потрібно довести, що 1 випадок зустрінеться лише скінченну кількість разів. Помітимо, що при проходженні 1 випадку при деякому значенні k величина d_{k-1} зменшується більше, ніж у $\frac{3}{4}$ раза, оскільки так зменшується значення $|b_{k-1}^*|^2$.

Однак величини d_i обмежені знизу деякою додатною константою, що залежить лише від решітки L. Точніше, якщо позначити через $m(L)$ квадрат довжини найкоротшого ненульового вектора L, то виконуються нерівності(3.11).

$$\mathrm{m(L)} \leq \left(\frac{4}{3}\right)^{\frac{i-1}{2}} \mathrm{d}_i^{\frac{1}{i}}, \mathrm{i} = 1, \dots, \mathrm{n}. \tag{3.11}$$

Теорема 4. Якщо L – решітка в Z^n з базисом $b_1, \dots, b_n$, причому

$$|b_i| \leq B, i = 1, \dots, n, \tag{3.12}$$

де $B \in R, B \geq 2$, то алгоритм побудови LLL-зменшеного базису робить $O(n^4 \log B)$ арифметичних операцій [6].

3.3 Теорія складності

З точки зору класичної математики проблеми в криптографії є тривіальними в тому сенсі, що вони можуть бути дозволені за кінцеве число спроб. Проте зведення до кінцевого числа випадків не має особливого сенсу у разі, коли число самих випадків не піддається обробці на практиці. І якщо ми не здатні розшифрувати деякі повідомлення в деяких часових межах, то ми можемо забути про усе інше, оскільки, коли пройде час, ситуація може повністю змінитися.

Тимчасова складність алгоритму є функція від довжини входу. Кажуть, що алгоритм має тимчасову складність f(n) тоді і тільки тоді, коли для усіх входів довжини n виконання алгоритму закінчується за не більше ніж f (n) кроків. Якщо n - число, то його довжиною буде число цифр або число розрядів в двійковому представленні n. Звичайно, для однієї і тієї ж проблеми можуть існувати як повільні, так і швидкі алгоритми. В деяких випадках можливе навіть необмежене прискорення.

Важким завданням є отримання нижніх оцінок складності, тобто показати, наприклад, що будь-який алгоритм для деякої проблеми має принаймні квадратичну оцінку складності.

Ясно, що тимчасова складність залежить від моделі алгоритмів, яку ми маємо на увазі. Число кроків буде менше, якщо на одному крок

виконуватиметься більше роботи. Проте фундаментальні поняття, такі, як поліноміальна тимчасова складність, значною мірою не залежать від вибору моделі алгоритмів. Звичайно, це стосується тільки моделей, вибраних розумно.

Щоб бути конкретнішим, ми виберемо машину Т'юрінга в якості нашої моделі алгоритмів. Машина Т'юрінга працює в дискретні моменти часу. У кожен момент часу вона може знаходитися в деякому внутрішньому стані (пам'яті), число яких кінцеве. Зчитуюче-записуюча голівка сканує букви, записані на стрічці, по одній в кожен момент в кожен момент часу. Кожна пара (q, a) визначає трійку (q1, a1, m), де q і q1 - це стани, a і a1 - букви і m означає одно з трьох значень : " наліво", " направо" і "на місці". Усе це означає, що в стані q, скануючи букву a, машина переходить в стан q1, записує a1 на місце a (можливо, a1 = a) і пересуває голівку у відповідності з m.

Якщо зчитуюче-записуюча голівка може "звалитися" із стрічки, тобто коли вимагається перейти наліво, а машина сканує крайній ліворуч квадрат стрічки, то до стрічки додається ліворуч новий порожній квадрат. Те ж саме робиться відносно правого кінця стрічки. Ця здатність нескінченного розширення зовнішньої пам'яті може розглядатися як вбудована особливість облаштування кожної машини Т'юрінга.

Сама стрічка може розглядатися і як потенційно нескінченна пам'ять, і як вхідний і вихідний канал. Формат входу-виходу описується таким чином. Машина починає свої обчислення, знаходячись в деякому початковому стані і прочитуючи крайню ліворуч букву цього вхідного слова. Обчислення закінчується, коли машина досягає деякого завершального стану. Тоді машина зупиняється і слово, що виявилося на стрічці, складе вихід. При зчитуванні виходу деякі допоміжні букви можуть ігноруватися.

Тепер ясно, що означає один крок обчислень. Ми можемо визначити функцію часу складності , пов'язану з машиною Тюрінга A, рівністю

fA(n)=max{m|A зупиняється після m кроків на вході w з |w| = n}.

Ми припускаємо для простоти, що A зупиняється, т. е. досягає завершального стану для усіх входів. Звичайно, це не так по відношенню до довільної машини Т'юрінга. Назвемо машину Т'юрінга A поліноміальною обмеженою, якщо існує многочлен p (n), такий, що fA (n) ≤ p (n) для усіх n. Позначення P використовується для класу проблем, що допускають рішення на поліноміальних обмежених машинах Т'юрінга.

Проблема називається (обчислювально) важко вирішуваною (іноді навіть практично нездійсненною), якщо вона не належить класу P . Легко вирішувані проблеми (тобто проблеми з P) утворюють декілька підкласів в P з очевидними визначеннями: завдання з лінійною, квадратичною, кубічною і іншою складністю. Неформальні поняття легкої проблеми означають, що степені многочленів малі, принаймні у вказаних вище межах.

Розглянута вище машина Т'юрінга являється детермінованою : внутрішній стан і буква на стрічці однозначно визначають її поведінку. Щоб підкреслити, що мається на увазі детермінована машина Т'юрінга, часто говорять про детерміновану тимчасову складність .

Недетермінована машина Т'юрінга має декілька можливостей своєї поведінки при зчитуванні букви. Відповідно вхідні слова призводять до декількох обчислень. Це можна представити як те, що машина робить припущення або використовує довільне число паралельних процесорів. Для кожного входу w розглядається найкоротше успішне обчислення S (w) (тобто обчислення, що веде до завершального стану). Функція тимчасової складності недетермінованої машини Т'юрінга A визначається як fA(n) = max{1, m| в s(w) m кроків для w з |w| = n} .

Розглядається пари (1, m), оскільки для деяких n, можливо, немає взагалі таких входів довжини n, які призводять до завершального стану.

Поняття поліноміальної обмеженої недетермінованої машини Т'юрінга і відповідний клас проблем, NP, визначаються в точності, як і в детермінованому випадку. Проблеми з класу P є легко вирішуваними, тоді як проблеми з NP

мають ту властивість, що легко вирішуваною виявляється перевірка: буде вдало вгадане рішення проблеми вірним або ні. Часову межу для недетермінованої машини Т'юрніга можна уявити собі як тимчасове обмеження на перевірку того, буде або немає вгадане рішення проблеми вірним. Відносно розкладання на множники не- відомо, чи лежить ця проблема в класі P, хоча, звичайно, вона з NP : досить вгадати співмножники і перевірити здогадку, вичисливши їх твір.

З визначень виходить, що P включається в NP, а відомою відкритою проблемою є питання: співпадають ці класи або ні, P = NP ? Незважаючи на те що це питання відкрите, можна вказати багато NP-повних проблем. Проблема являється NP-повною якщо і тільки якщо вона з NP і, крім того, являється NP-важкою, тобто будь-яка проблема з NP може бути зведена до неї за поліноміальний час. Звідси витікає, що P = NP тоді і тільки тоді, коли деяка NP-повна проблема належить класу P . В цьому випадку довільна проблема з NP може бути вирішена за поліноміальний час, оскільки спершу за поліноміальний час вона може бути зведена до даної NP-повної проблеми, яка у свою чергу по припущенню може бути вирішена за поліноміальний час. Ясно, що композиція двох поліномів знову буде поліномом.

Загальне переконання полягає в тому, що P ≠NP. Таким чином, NP - повні проблеми розглядаються як важко вирішувані. Окрім NP, терміни " важка" і " повна" використовується в тому ж сенсі і у зв'язку з іншими класами проблем.

Те, що конкретна проблема являється NP-важкою, доводиться тим, що деяка проблема, про яку вже відомо, що вона являється NP-важкою, може бути зведена за поліноміальний час до даної проблеми. Якщо ми хочемо показати, що ця проблема являється NP-повною, то ми повинні показати також, що вона з NP [2].

3.4 Висновки до розділу

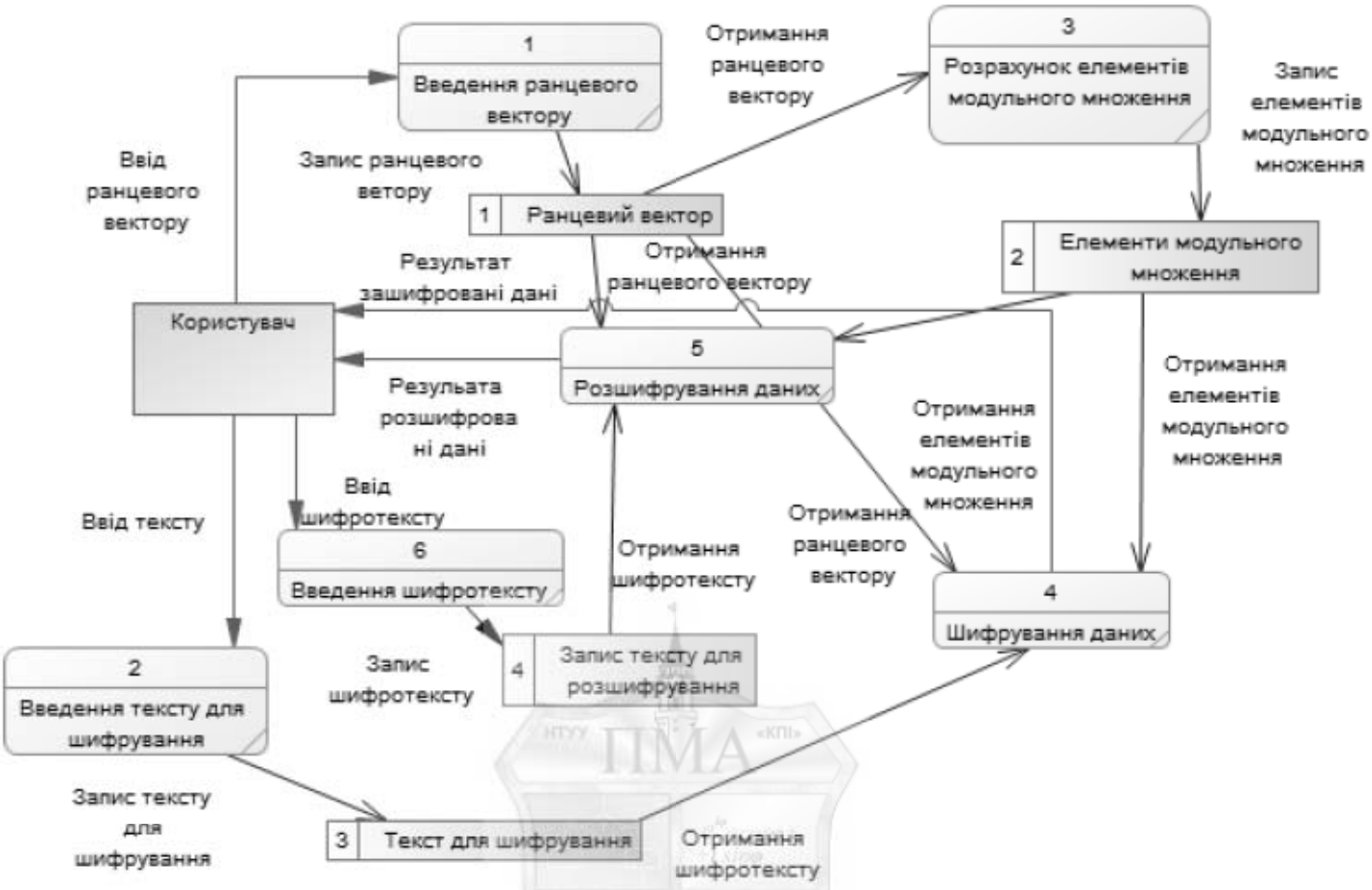
У даному розділі розроблено математичне забезпечення для системи, що проектується. Розглянуто теорію складності алгоритмів та основні поняття цілочисельних решіток. Також було більш детально розглянуто основні властивості LLL-зменшеного базису та алгоритм побудови зменшення базису для цілочисельної решітки. Побудовано зручний алгоритм для програмної реалізації обраного методу криптоаналізу системи шифрування Меркла-Хелмана. Доведено, що алгоритм побудови LLL-базису завершую свою роботу за скінченну кількість кроків.

4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Структура програми шифрування

Програма написана на мові програмування Java, інтерфейс спроектований за допомогою бібліотеки Swing, та має наступну структуру виконуваних процесів зображених за допомогою DFD-1 (Рис.1), та DFD-0 (Рис.2).

Рисунок 1 - DFD-0 системи



. Рисунок 2 - DFD -1 системи

4.2 Функціональна частина систем

У системі шифрування реалізовані наступні функціональні частини:

1. Зчитування даних ранцевого вектора, що вводиться користувачем, перевірка коректності даних (коректності вводу формату цілого додатного числа, та виконання умови надзростання для ранцевого вектору) відбувається в окремому потоці, що працює паралельно з основним.

2. Генерація елементів модульного множення для введеного ранцевого вектору.

3. Шифрування введеного користувачем тексту на основі ранцевого вектору, що отримано з введеного користувачем вектору шляхом модульного множення(далі - перемішаного ранцевого вектору), вивід на екран зашифрованого тексту за допомогою алгоритму Меркла-Хеллмана.

4. Розшифрування зашифрованого за допомогою перемішаного ранцевого вектору тексту використовуючи надзростаючий вектор, та операцію зворотного модульного множення для компонент шифрованого тексту. У разі вводу некоректного зашифрованого тексту про це буду сповіщено користувача повідомленням про помилку.

У програмній реалізації криптоаналізу за допомогою LLL-алгоритму розроблені наступні функціональні частини:

1. Ортогоналізації системи векторів за допомогою методу Грама-Шмідта.

2. Алгоритм отримання LLL зменшеного базису цілочисельної решітки.

3. Використання LLL зменшеного базису для вирішення задачі про укладку ранцю.

4.3 Формат вихідних даних

Вихідні дані системи шифрування складаються з ранцевого вектора, на основі якого проводиться шифрування та розшифрування даних, та безпосередньо текст і шифротекст.

До формату ранцевого вектора висуваються наступні обмеження:

1. Вектор, довжиною від 10 до 25 компонент, числового формату.

2. Невід'ємне ціле число, що не перевищує 8знаків.

3. Для ранцевого вектора необхідне виконання умови надзростання, що кожне наступне число вектору більше суми усіх попередніх.

Формат тексту - довільний набір символі кількість яких не перевищує 625.

До формату шифротексту висуваються наступні вимоги:

1. Цілі невід'ємні числа, що йдуть через пробіл.

2. Цей набір чисел являє собою зашифрований текст у системі Меркла-Хеллмана, вхідний текст для якого не перевищує довжину в 625символів.

Вихідні дані системи для криптоаналізу складаються з ранцевого вектору A - вектор цілих додатних чисел, та маси ранцю K - цілого додатного числа.

4.4 Формат результуючих даних

Результуючі дані системи шифрування представлені у вигляді зашифрованого та розшифрованого текстів за допомогою системи Меркла-Хеламана.

Формат зашифрованого тексту - цілі невід'ємні числа, що йдуть через пробіл, кількість чисел залежить від довжини введеного тексту і довжини ранцевого вектору.

Формат розшифрованого тексту - довільна послідовність символів, довжиною не більше ніж 625.

Результуючими даними системи для криптоаналізу є вектор X, компоненти якого числа 0 або 1, для яких виконується умова:

$$\sum_{i=1}^{n} x_i a_i = K \qquad (4.1)$$

де K та a_i - вихідні дані.

4.5 Тестування розроблених програмних засобів

4.5.1 Введення ранцевого вектору

Користувач може ввести ранцевий вектор вручну або обрати його за замовченням (Рис.3-Рис.4).

Рисунок 3 - Введення ранцевого вектору вручну

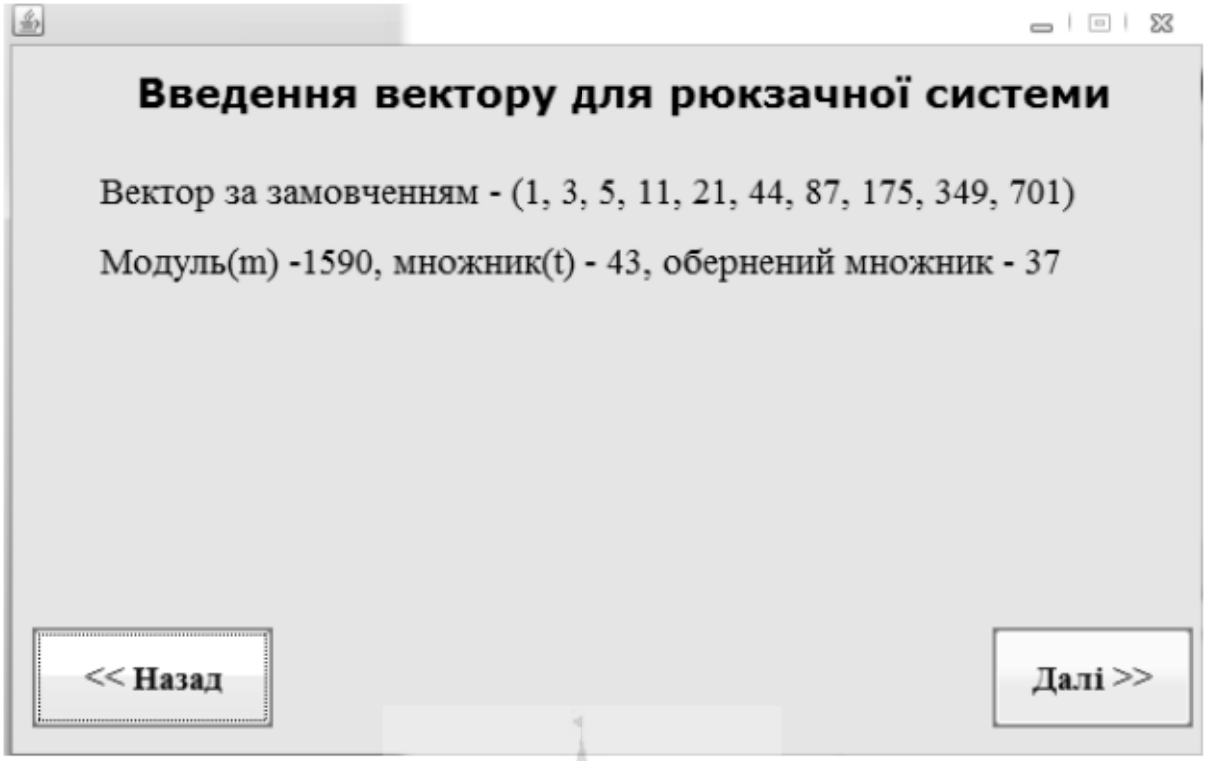


Рисунок 4 - Використання ранцевого вектору за замовченням

4.5.2 Шифрування даних

Шифрування відбувається на основі відкритого ключа (перемішаного ранцевого вектору за допомогою модульного множення) розробленої асиметричної системи Меркла-Хеллмана (Рис. 5).

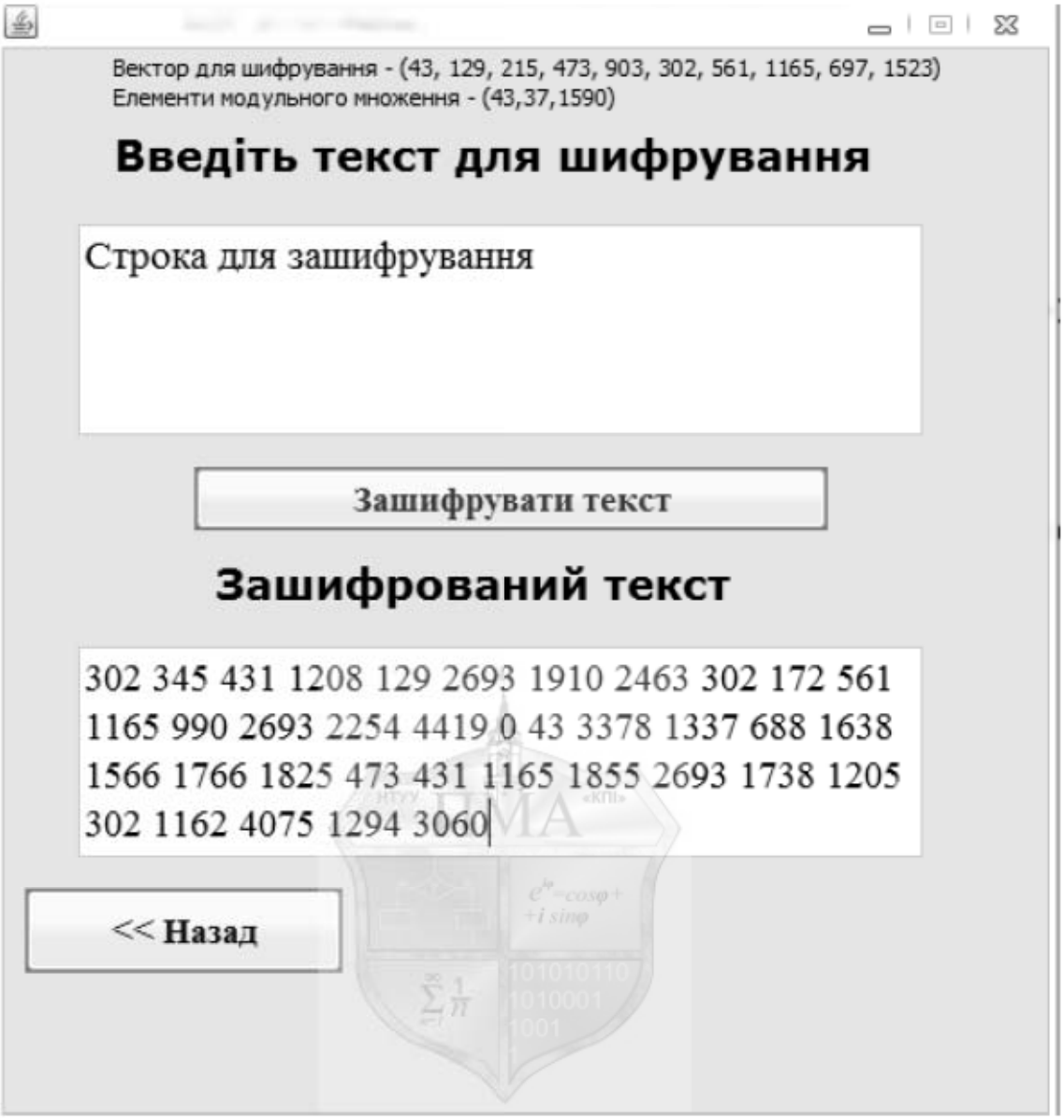


Рисунок 5 - Шифрування даних

4.5.3 Розшифрування даних

Розшифрування ведеться на основі закритого ключа (Рис. 6). Отримати розшифровані дані може тільки людина, що має закритий ключ, а саме - елементи модульного множення.

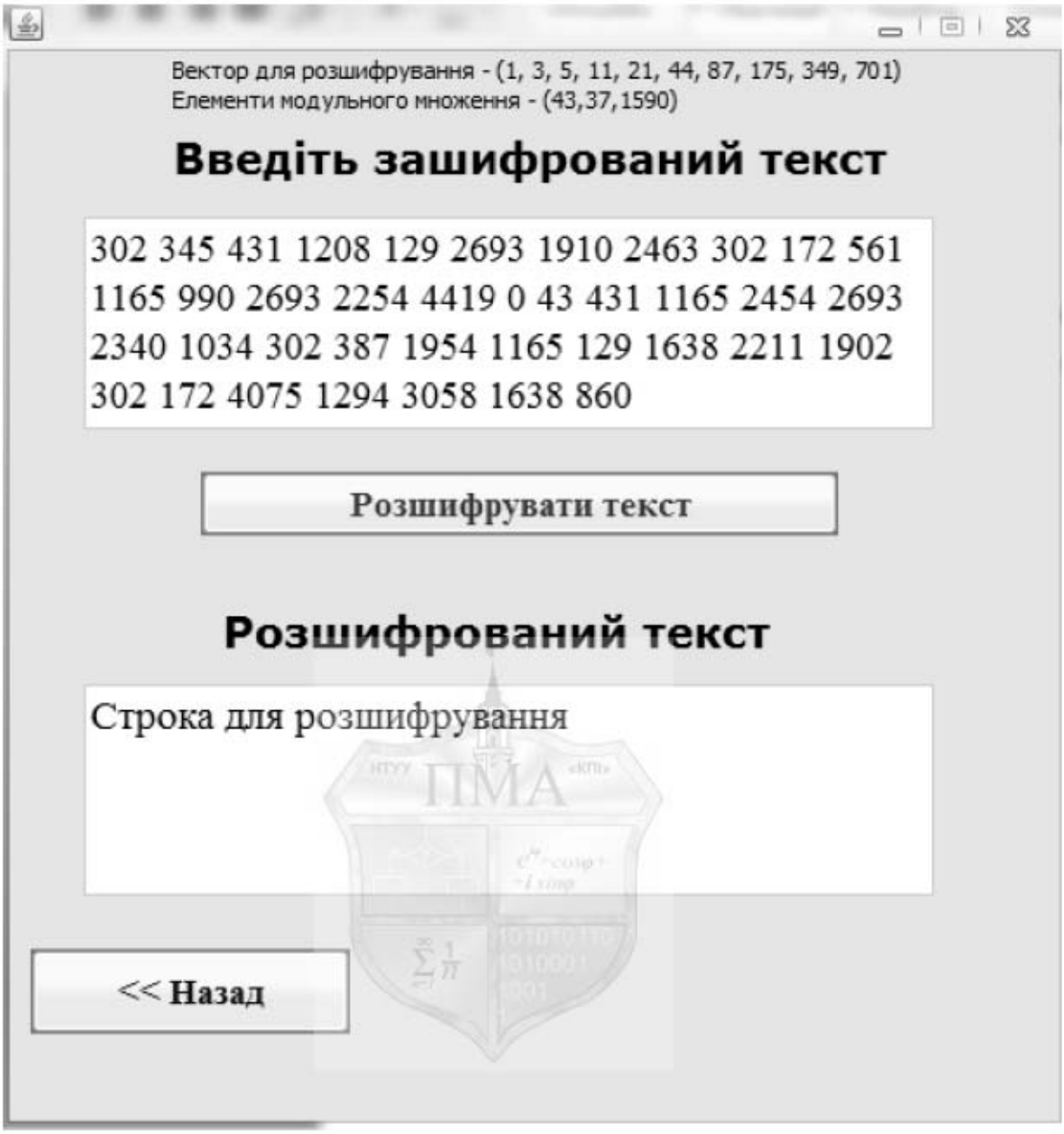


Рисунок 6 - Розшифрування даних

4.5.4 Криптоаналіз LLL-методом.

Робота програми заключається в вирішенні задачі про укладку ранцю, що має розв'язок, для довільних заданих ранцевого вектору та маси ранцю. Спочатку отримується система векторів, на якій будується цілочисельна решітка (Рис.7),

після чого знаходиться LLL-зменшений базис даної решітки і потім вже результуючий вектор, який є розв'язком даної задачі про укладку ранцю (Рис.8).

```
Базис цілочисельної решітки:
1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 86.0
0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 258.0
0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 430.0
0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 946.0
0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1806.0
0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 604.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1122.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 2330.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1394.0
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 3046.0
0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 2342.0
```

Рисунок 7 - Базис цілочисельної решітки

```
LLL-зменшений базис:
-0.5, -0.5, 0.5, 0.5, 0.5, -0.5, 0.5, 0.5, -0.5, 0.5, 0.0
1.0, -2.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
-1.5, -0.5, 0.5, -0.5, -0.5, 0.5, -0.5, -0.5, 0.5, -0.5, 0.0
0.0, -2.0, -1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.0, 0.0, -2.0, -1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
0.5, 0.5, 0.5, -0.5, 0.5, 0.5, 0.5, 0.5, -0.5, 0.5, 2.0
-0.5, -0.5, 0.5, -0.5, -0.5, -1.5, 0.5, -0.5, 0.5, -0.5, 0.0
-1.0, 0.0, 0.0, 0.0, 0.0, 0.0, -2.0, 1.0, 0.0, 0.0, 0.0
0.5, -0.5, -0.5, -0.5, -0.5, 0.5, -0.5, -0.5, -1.5, 0.5, 0.0
-0.5, 0.5, -0.5, 1.5, 0.5, -0.5, -0.5, -2.5, -0.5, 0.5, 0.0
0.5, -0.5, 1.5, 0.5, 0.5, -0.5, 0.5, -1.5, -1.5, -2.5, 0.0
Густина ранцю - 0.9458321701912503
Результат:
Ранцевий вектор:  43.0 129.0 215.0 473.0 903.0 302.0 561.0 1165.0 697.0 1523.0
Результуючий вектор:  1 1 0 0 0 1 0 0 1 0
```

Рисунок 8 -LLL-зменшений базис, та результуючий вектор

4.6 Результати проведення криптоаналізу

Результати тестування LLL методу зменшення базису для розв'язку задачі про укладку ранцю приведені у таблиці 1.

Таблиця 1 - Результат проведення криптоаналізу

№	Ранцевий вектор, A	Маса, K	Густина d(A)	Розв'язок, X
1	(43,129,215,473,903,302,561, 1165,69,7,1523)	1171	0,946	(1, 1, 0, 0, 0, 1, 0, 0, 1, 0)
2	(1,3,5,11,21,44,87,175,349, 701)	733	1,058	(0, 0, 0, 1, 1, 0, 0, 0, 0, 1)
3	(251,346,129,321,217,281,391,456,251,467,25,73,113,249,375,218,453,122,197,234)	1361	2,255	(1, 0, 0, 0, 0, 1, 0, 0, 0 ,1, 0, 0, 1, 1, 0, 0, 0, 0,0, 0)
4	(251,346,129,321,217,281,391,456,251,467,25,73,113,249,375,218,453,122,197,241,354,564,621,256,432,876,121,245,111,454, 123,234)	1238	3,274	—
5	(251,3466,129,321,217,281,391, 456,251,467,25362,73,1138,249,375,218,4538,122,197,241,35456,564,62112,256,432,876,121, 245,11132,454,12345, 234)	1238	2,01	(0, 0, 1, 0, 0, 0,1, 0, 1, 1, 0,0, 0 ,0, 0, 0, 0, 0, 0 ,0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
6	(2510,346,3295,321,217,2891,391,7560,467,25,73,113,249,375, 218,453,1223,197)	1361	1,4	(0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0)
7	(251,466,129,321,217,281,391,456,251,467,5362,73,1138,249,375,218,538,122,197,241,35456, 564,2112,256,432,876,121,245, 11132,454,123,234)	1238	2,11	—
8	(12063,16084,32168,64336,128672,261365,522730,104981,2135151,1001230,2239699,570987, 1612431,3715430,88512,1930186,1379432,2606043,825278,3464243)	378409	0,92	(1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

Продовження таблиці 1

№	Ранцевий вектор, A	Маса, K	Густина d(A)	Розв'язок, X
9	(1417203, 2729428, 5511345, 11022690, 23620050, 51176775, 21363025, 31650871, 11180166, 22990200, 25247245, 16954021, 23042827, 44353515, 15799811, 15275570, 28609063, 45449643)	41991274	0,703	(1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0)
10	(1361, 2722, 5444, 10888, 23137, 44913, 91187, 183735, 372914, 763521, 411023, 768967, 255873, 656012, 1049351, 156557, 477795, 374444, 120107, 1239187)	1113303	0,988	(0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0)

З таблиці видно, що вдалість проведеного криптоаналізу залежить від густини ранцевого вектора, чим вона менша, у більшості випадків, тим більший шанс на вдало проведений результат роботи алгоритму. Насамперед слід зазначити, що для векторів, що є "перемішаними", за допомогою модульного множення, надзростаючими векторами (№1, 8,9,10 з таблиці 1) густина дорівнює не більше ніж 0,988, для протестованих зразків. Оскільки для проектування реальної системи ми мусимо використовувати саме такі вектори, з невеликим показником густини, то для них LLL алгоритм показує високу ймовірність вдалого криптоаналізу.

4.7 Висновки до розділу

У даному розділі було спроектовано програмне забезпечення для системи асиметричного шифрування Меркла-Хеллмана, за допомогою якого можна шифрувати та розшифровувати текст на основі задачі про укладку ранця, побудованій на введеному ранцевому векторі. Ранцевий вектор обирається за замовченням або вводиться в ручну з довжиною у межах від 10 до 25, ввід строго форматується системою. Користувач , який ввів текст, отримує шифротекст у вигляді послідовності чисел через пробіл, та може отримати розшифрований текст маючи шифротекст.

Також було розроблено програмне забезпечення для криптоаналізу даної системи, що полягає у вирішенні задачі про укладку ранцю LLL-методом зменшення базису цілочисельної решітки. Проведено тестування створених програм, та встановлено залежність між можливістю вдалого криптоаналузу та вхідних даних системи. За аналізом результатів було встановлено, що для ранцевий векторів густина яких не перевищує 0,988 ймовірність проведення вдалого криптоаналізу дуже висока, враховуючи, що для створення системи використовуються вектори саме з такою густиною, то LLL-метод дуже вдало показує себе для вирішення поставленої задачі.

ВИСНОВКИ

У роботі розглянуто побудову та структуру системи асиметричного шифрування Меркла-Хеллмана. Переглянуто два методи криптоаналізу даної системи, та обрано з поміж них LLL-алгоритм через його швидкодію та безперечну універсальність у порівнянні з підходом Шаміра для знаходження елементів модульного множення.

Було спроектовано програмне та математичне забезпечення для розробленої системи шифрування, та її криптоаналізу, що відповідає поставленій задачі. Система шифрування забезпечує шифрування та розшифровування тексту на основі задачі про укладку ранця, побудованій на введеному ранцевому векторі. Ранцевий вектор обирається за замовченням або вводиться в ручну з довжиною у межах від 10 до 25, при строго форматованому системою вводі. Програма для проведення криптоаналізу вирішує задачу про укладку ранцю для введеного ранцевого вектора та відповідної маси ранцю.

Проведено тестування створених програмних засобі, та аналіз отриманих результатів. Встановлено залежність між можливістю вдалого криптоаналізу та вхідними даними системи (густиною ранцевого вектора). За результатами було встановлено, що для ранцевий векторів густина яких не перевищує 0,988 ймовірність проведення вдалого криптоаналізу дуже висока, враховуючи, що для створення системи використовуються вектори саме з такою густиною, то LLL-метод дуже вдало показує себе для вирішення поставленої задачі.

Розроблені програмні засоби можна використовувати у навчальних цілях для наочної демонстрації роботи системи з відкритим ключем Меркла-Хеллмана, та її криптоаналізу з використанням LLL-метод зменшення базису цілочисельної решітки. Також програмну реалізацію LLL-методу можна використовувати для інших цілей, наприклад факторизації многочленів. Не виключене і використання

системи для шифрування повідомлень в межах невисокої конфіденційності, розуміючи той факт, що вона не є стійкою до криптоаналізу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Шнайер Брюс Прикладная криптография. – М.: ТРИУМФ, 2003. – 816с.
2. Саломаа А. Криптография с открытым ключом. — Москва: Мир, 1995. —318 с.
3. Shamir A. A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem // Advanced in Cryptology: Proceedings of Crypto 82 — Plenum Press, 1983. — P. 270-288
4. Маховенко Е.Б. Теоретико-числовые методы в криптографие. — Москва: Гелиос АРВ, 2006. — 320 с.
5. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии. — Москва: МЦНМО, 2003.—328 с.
6. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovasz. Factoring Polynomials with Rational Coefficients. Mathematische Annalen. —1982. — P. 515-534.