МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Західноукраїнський національний університет

Факультет комп’ютерних інформаційних технологій

Кафедра кібербезпеки

БАЙДА Максим Олегович

**АЛГОРИТМ ФАКТОРИЗАЦІЇ НА ОСНОВІ ЕЛІПТИЧНИХ КРИВИХ/ THE FACTORIZATION ALGORITHM BASED ON ELLIPTIC CURVES**

Спеціальність 125 – Кібербезпека

Освітньо-професійна програма – Кібербезпека

Кваліфікаційна робота

Виконав студент групи КБм-21

М.О. Байда

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Науковий керівник:

к.т.н., доцент, С.В. Івасьєв

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

кваліфікаційну роботу допущено

до захисту

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_р.

Завідувач кафедри

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Яцків

Тернопіль 2021

ЗМІСТ

|  |  |
| --- | --- |
| Вступ…………………………………………………………………………….. | 3 |
| 1 Теорія еліптичного шифрування……………………….……………….. | 5 |
| 1.1 Асиметричне шифрування на еліптичних кривих…………..…….. | 5 |
| 1.2 Криптографічні алгоритми ………………………………………… | 8 |
| 1.2 Ефективні алгоритми генерації еліптичних кривих………………. | 13 |
| 2 Дослідження можливостей криптоаналізу………………………………….. | 18 |
| 2.1 Аналіз існуючих підходів до факторизації……………………….... | 18 |
| 2.2 Криптоаналіз асиметричних алгоритмів шифрування……….…… | 33 |
| 2.3 Розробка системи криптоаналізу для асиметричних криптосистем…………………………………………………………… | 36 |
| 3 Факторизація на основі еліптичних кривих………………………………… | 42 |
| 3.1 Аналіз можливостей криптографії на основі еліптичних кривих .. | 42 |
| 3.2 Субекспоненційні алгоритми факторизації………………………. | 47 |
| 3.3 Алгоритм факторизації на основі еліптичних кривих | 50 |
| Висновки……………………………………………………………………….... | 59 |
| Список використаних джерел…………………………………………………. | 60 |
|  |  |

# ВСТУП

**Актуальність роботи.** Сьогодні криптосистеми на еліптичних кривих використовуються в TLS, PGP і SSH, найважливіших технологіях, на яких базуються сучасний веб і світ ІТ також це стосується Bitcoin та інших криптовалют.

До того, як ECC стала популярною, майже всі алгоритми з відкритим ключем ґрунтувалися на RSA, DSA і DH, альтернативних криптосистемах на основі модулярної арифметики[1]. RSA та інші алгоритми що базуються на обчислювальній складності алгоритму факторизації, як і раніше, популярні, і часто використовуються разом ECC. Переваги стікості шифрів на основі еліптичних кривих наштовхують на необхідність розвитку досліджень можливості арифметики еліптичних кривих, щодо факторизації.

Проблема факторизації безпосередньо пов'язана з визначенням криптостійкості RSA, яке базується на припущенні, що не існує швидких алгоритмів факторизації, які за короткий час дозволили б зламати код, а якщо через деякий час і вийде це зробити, то дані втратять свою актуальність.

**Мета роботи** полягає в розробці алгоритму факторизації на основі еліптичних кривих та досліджені використання еліптичних кривих в сучасних системах криптоаналізу .

Для досягнення даної мети ставились наступні завдання:

* Дослідити теорію еліптичного шифрування.
* Проведести оцінку асиметричного шифрування на еліптичних кривих.
* Дослідити основні етапів алгоритмів квадратичного решета, ферма подібних алгоритмів для факторизації чисел.
* Проведести дослідження існуючих новітніх підходів до факторизації.
* Проведено дослідження та розробку системи для крипто аналізу.
* Дослідити математичні основи алгоритмів для ефективної генерації еліптичних кривих.
* Дослідити можливості криптографії та крипто аналізу на основі еліптичних кривих.
* Розробити програмний засіб для факторизації на основі еліптичних кривих.

**Об’єкт дослідження** - процеси програмного та апаратного опрацювання багато-розрядних чисел.

**Предмет досліджень** – алгоритми факторизації багаторозрядних чисел та методи криптоаналізу.

**Наукова новизна одержаних результатів** визначається наступним чином:

- досліджено математичні основи алгоритмів для ефективної генерації еліптичних кривих. досліджено можливості криптографії та криптоаналізу на основі еліптичних кривих.

- розроблено алгоритм факторизації на основі еліптичних кривих, що дозволяє проводити факторизацію багаторозрядних чисел використовуючи декілька еліптичних кривих.

**Практична цінність одержаних результатів** полягає в тому, щорозроблено програмний засіб для факторизації на основі еліптичних кривих. Проведена програмна реалізація алгоритму Ленстри для факторизації багаторозрядних чисел на основі еліптичних кривих.

Публікації та апробація до магістерської роботи. За результатами наукових досліджень, проведених у магістерській роботі, підготовлено тези доповіді на проблемно-науковій міжгалузевій конференції «Кібербезпека та комп’ютерно-інтегровані технології» (КБКІТ – 2021, додаток А).

# 1 ТЕОРІЯ ЕЛІПТИЧНОГО ШИФРУВАННЯ

1.1 Асиметричне шифрування на еліптичних кривих

Перед тим, як показати функціонал обробки, хотілося б розглянути, яке завдання вирішує шифрування на еліптичних криві і розповісти коротку теорію тим, хто стикався з цією темою.

Проблема проста – двоє користувачів хочуть безпечно обмінятися інформацією. Зазвичай у літературі цих користувачів називають Боб і Аліса, хоча реально це може бути: бухгалтер, який складає електронну звітність та податкову службу; клієнт банку, що оплачує покупки електронною карткою та банківська система, що фіксує та обробляє транзакції; трейдер, що купує bitcoin і ферму, що отримує його запит і т.д.

Суть криптографічної захисту полягає в тому, що якщо злодій перехопить повідомлення, що відсилається (наприклад, через сумнівний Wi-Fi у разі оплати через телефон або за допомогою вірусної програми, що перехоплює повідомлення, що відсилаються з комп'ютера), то він не зможе її розшифрувати та змінити. А то представте, що було б без захисту - клієнт надсилає інформацію банку про переказ 100 рублів іншому клієнту, злодій перехоплює дане повідомлення і змінює в ньому суму зі 100 рублів на 100 тис. рублів, так само змінює рахунок одержувача на свій і надсилає це повідомлення далі до Банку, Банк без перевірки проводити транзакцію... Від подібних дій зловмисників рятує криптографічний захист.

Працює криптографія дуже цікаво – у кожної сторони обміну повідомлення існує відкритий (H) та закритий ключ (d). Причому відкритий ключ бачать всі учасники (він як номер банківської картки або номер телефону), закритий ключ – це як би секретний пароль, який знає тільки користувач.

Є два основні алгоритми використання криптографічних ключів:

1) формується загальний секретний ключ за даними відкритих ключів відправника та одержувача та закритого ключів відправника. У цьому випадку відбувається шифрування повідомлення за допомогою спільного секретного ключа з перетворенням повідомлення на послідовність символів, що не читається. Зловмисник, який перехопив повідомлення, знаючи лише відкриті ключі відправника та одержувача та не знаючи закритого ключа, не зможе його розшифрувати. Отримувач зможе обчислити загальний секретний ключ і розшифрує повідомлення.

2) формується цифрова підпис повідомлення з використанням відкритого та закритого ключа відправника. У цьому випадку повідомлення, що пересилається, складатиметься з тексту повідомлення та цифрового підпису (у вигляді послідовності цифр), сформованого за текстом повідомлення. Таке повідомлення може пересилатися у відкритому вигляді, оскільки злодій, який перехопив повідомлення, зможе його прочитати, але при спробі зміни повідомлення перестане відповідати цифровому підпису і сторона, яка приймає, перевіривши, відхиляє таке повідомлення, як не відповідне цифровому підпису.

Таким чином і працює криптографічний захист. Як відомо з описаних алгоритмів - реалізації описаного функціоналу необхідні відкриті та закриті ключі, тому давайте коротко розглянемо теорію шифрування на еліптичних криві.

Сучасне шифрування базується на пошуку координат на кривій. Криві шифрування трохи складніші, зустрічаються в повсякденному житті і називаються еліптичними кривими. Вони мають вигляд:



Оскільки для криптографічних цілей достатньо колекції потрібних точок (а не всі точки кривої), криву обмежують, додаючи модуль:



Криптографія реалізується за допомогою специфічних особливостей цього класу кривих.

Перше - це те, що для еліптичних кривих існує правило: якщо провести пряму, що перетинає 3 точки кривої, то сума точок перетину дорівнюватиме 0.



Це ж рівняння можна уявити як:



і переформулювавши правило: результатом складання двох точок графіка буде зворотна величина третьої точки. Це рівняння працює навіть якщо одна з точок стане дотичною (як це виглядає графічно, дивіться на логотипі публікації ;-) ).

Друге – те, що еліптичні криві мають одну цікаву особливість, звану циклічність. Полягає воно в тому, що виробляючи послідовні множення точки на змінну, значення координат точок через ряд множень почне повторюватися. І тут ми отримуємо важливі параметри:

- Базова точка еліптичної кривої ( ***G*** ), тобто. це перша точка, обчислена після крапки з координатою (0, 0).

- Порядок кривої ( ***n*** ), тобто. значення змінної, починаючи з якої починаються повторення координат точок.

Так ось, відкриті та закриті ключі формуються на однаковій еліптичній кривій, для якої відома Базова точка ( ***G*** ), Порядок кривої ( ***n*** ) та інші параметри (які опускаю, щоб не ускладнювати опис). Ці параметри є постійними для кожної окремої еліптичної кривої і відомі всім учасникам обміну.

Закритий криптографічний ключ ( ***d*** ) – це випадкове число діапазоні від 1 до порядку кривої ( ***n*** ), тобто. до кількості додавань базової точки ( ***G*** ), після яких точки починають повторюватися.

Відкритий криптографічний ключ ( ***H*** ) – це точка еліптичної кривої, яка виходить після множення Базової точки ( ***G*** ) на ***d*** (значення закритого криптографічного ключа).

1.2 Криптографічні алгоритми

Широке поширення інформаційних технологій, які включають взаємодію безлічі абонентів, різноманітної інформації та широкого спектру телекомунікаційних послуг, передбачає розвиток засобів, що дають можливість гарантувати належним чином цілісність оброблюваної інформації, її автентичність і, при необхідності, конфіденційність. Вирішити це завдання в даний час можливо лише, застосовуючи криптографічні алгоритми обчислення та перевірки цифрового підпису та алгоритми шифрування. Причому у разі, коли в інформаційному процесі бере участь дуже обмежена кількість абонентів та комунікації між ними обмежені, питання ідентичності та автентичності можна вирішити, обчислюючи імітівставку, як рекомендує ГОСТ 28147-89. Але тут є ще одне обмеження – учасники інформаційного обміну мають повністю довіряти один одному і тільки таким чином виключити можливість несанкціонованої передачі секретного ключа, який за складним протоколом пересилається всім учасникам обміну, та можливість зміни інформаційної посилки з обчисленням на неї нової імітовставки. У разі великої кількості учасників інформаційної взаємодії та розгалужених зв'язків застосування для встановлення ідентичності та автентичності інформаційних документів методу обчислення імітівставки вкрай складно та неефективно через вкрай складну організацію протоколів розсилки секретних ключів та неефективності умови “загальної та повної” довіри. Єдиний розумний спосіб досягти коректної реалізації процедури ідентифікації та аутентифікації надають криптографічні алгоритми обчислення та перевірки цифрового підпису, що базуються на криптографічних перетвореннях з несиметричними ключами. Ці алгоритми ідеально підходять для великих інформаційних мереж, оскільки їхнє застосування не пов'язане з попереднім обміном секретною інформацією і, отже, повністю знімається складне завдання управління секретними ключами. Тому не викликає сумніву, що розробка та прийняття в 1994 р. російського стандарту цифрового підпису Р 34.10-94 та пов'язаного з ним стандарту обчислення функції хешування Р 34.11-94 було своєчасним і зіграло і продовжує відігравати важливу роль у вирішенні завдань забезпечення цілісності інформації її походження. Згадані вище стандарти були адаптовані як стандарти України відповідно до ГОСТ 34.310-95 та ГОСТ 34.311-95 і діють в даний час. Чинний стандарт цифрового підпису використовує надійні алгоритми обчислення та перевірки цифрового підпису. Його стійкість заснована на неймовірній складності розв'язання так званої задачі дискретного логарифмування у мультиплікативній групі простого кінцевого поля великого порядку. Стандарт досить простий у реалізації та за своїми алгоритмічними та криптографічними властивостями знаходиться на рівні аналогічних нині діючих зарубіжних стандартів: національного стандарту США DSS FIPS 186-2 та низки галузевих та міжнародних стандартів, що використовують алгоритм DSA, закріплений у стандарті США, наприклад, стандарту ANSI X9 .30-1, що використовується в банківській сфері, та стандарту ISO/IEC 14888. При збереженні нинішніх тенденцій розвитку обчислювальної техніки та алгоритмічної теорії він може безпечно використовуватися у відповідальних застосуваннях, принаймні до 2005 р., а менш відповідальних – аж до 2010 р. Проте, на нашу думку, настав час готувати заміну цього стандарту. Причин цього досить багато і пов'язані як із особливостями самого стандарту, і світовими тенденціями розвитку інформаційних технологій. Чинний стандарт не позбавлений недоліків алгоритмічної та структурної властивості. Недоліки алгоритмічного характеру. 1. Використовується невдалий спосіб створення основних параметрів алгоритму, що має тонкі місця. Досвід сертифікації показує, що це часто призводить до помилок під час реалізації алгоритму та ускладнює процес сертифікації. Цей спосіб позбавляє стандарт гнучкості у виборі параметрів безпеки стандарту. 2. Зафіксований у стандарті спосіб генерації параметрів призводить до того, що в даний час можна використовувати лише одне значення розміру ключа 1024 біта.

Передбачене стандартом значення 512 біт зараз використовувати не можна з міркувань безпеки. Стандарт не дозволяє використовувати проміжні значення, що було б зручним для багатьох застосувань, оскільки розмір ключа залежить швидкість виконання обчислень. У той же час не можна і збільшити це значення, що, безсумнівно, буде потрібно після 2005 року у відповідальних застосуваннях. Усунути цей недолік без істотної переробки стандарту не можна. 3. Використано невдалий спосіб перевірки псевдопростоти цілих чисел. Нескладно побудувати складове число, яке у такий спосіб буде визнано псевдопростим. Цю особливість теоретично можна використовуватиме компрометації підпису. Структурні вади. 1. Великий недолік чинного стандарту – відсутність нормованого алгоритму генерації секретних постійних та разових ключів. 2. Стандарт прив'язаний до конкретної функції хешування ДЕРЖСТАНДАРТ 34.311-95, в якій, як відомо, використовується стандарт шифрування даних ГОСТ 28147-89. Це ускладнює сертифікацію реалізацій стандарту, подорожчає її та вимагає придбання довгострокових ключів у уповноваженої організації. На нашу думку, обчислення функції хешування має бути самостійним алгоритмом, який не має посилань до інших стандартів.

Відсутня формалізація вхідних та вихідних даних, що ускладнює використання стандарту у великих різнорідних мережах. В даний час стандарт практично використовується лише у корпоративних мережах, створених одним розробником. Це ускладнює використання цифрового підпису у великих мережах, створюваних різними розробниками, та перешкоджає створенню загальнонаціональної системи використання та сертифікації цифрового підпису. Тепер про причини необхідності розробки нового стандарту, які мають технологічний характер та не пов'язані безпосередньо з чинним стандартом. 1. Швидкий розвиток обчислювальної техніки та розробка ефективних методів криптоаналізу, що застосовуються саме до алгоритмів того типу, що використовується в чинному стандарті, зажадали збільшити довжину ключа до 1024 біт, а до 2005р. безпечна довжина ключа буде щонайменше 1600 біт. Це ускладнює використання стандарту та зменшує його швидкодію. 2. Алгебраїчні структури, що використовуються у чинному стандарті, погано підходять для реалізації на мікропроцесорах, що використовуються у смарт-картках. Відомо, як широко такі пристрої використовуються в різних цілях. 3. Для забезпечення довготривалої інформаційної безпеки від малоймовірного, але все ж таки можливого, прориву в теоретичній галузі, який може спричинити повну дискредитацію чинних алгоритмів, бажано мати алгоритми, які використовують інші теоретичні принципи. Новий стандарт на нашу думку повинен мати наступні властивості: 1) забезпечувати дуже високу стійкість і ґрунтуватися на нових теоретичних принципах, що виключають застосування існуючих ефективних методів криптоаналізу; 2) бути гнучким щодо вибору параметрів безпеки та досить простим у реалізації; 3) допускати просту апаратну реалізацію; 4) використовувати нормований, прийнятий нормативним документом, наприклад, ГОСТом, датчик випадкових чисел для отримання секретних ключів; 5) не бути прив'язаним до конкретної функції хешування; 6) забезпечувати однозначне подання вхідних та вихідних даних. У першу чергу необхідно вибрати структуру алгебри, найбільш підходящу для реалізації алгоритмів нового стандарту. Ми вибрали групу точок еліптичної кривої над кінцевим полем характеристики 2. Цей вибір визначено як власним багаторічним досвідом роботи з цим об'єктом, і очевидною світової тенденцією дедалі ширшого впровадження алгоритмів цього у криптографічні протоколи. Вперше криптографічні алгоритми у таких групах було запропоновано 1985 р. Віктором Міллером [1] і Нілом Коблицем [2] і тривалий час вважалися екзотикою, непридатною практичного застосування. Обчислення порядку еліптичної кривої здавалося дуже складним завданням, тому спочатку пропонувалося використовувати спеціальні види еліптичних кривих, де це завдання вирішувалося досить просто. Згодом багато хто з цих типів кривих виявився непридатним для криптографічних застосувань, наприклад, суперсингулярні криві. Тому дуже важливим є суто теоретичний результат Р. Схофа [3], з якого випливає, що обчислення порядку еліптичної кривої є завданням поліноміальної складності, тобто дуже просте завдання.

Незабаром було побудовано ефективні практичні алгоритми обчислення порядку еліптичних кривих. Довгий час залишалося незрозумілим, як співвідносяться між собою задача дискретного логарифмування групи точок еліптичної кривої і аналогічне завдання у полі визначення кривої. У 1993 р. А. Менезес, Т. Окамото та C. Венстон показали, що в принципі перше завдання зводиться до другого, але умови відомості легко контролюються і завжди можна вибрати еліптичну криву, для якої це зведення не дає субекспоненційного алгоритму розв'язання задачі дискретного логарифмування у групі точок еліптичної кривої (знамените MOV-condition). Можна вважати, що з цього моменту розпочалася активна робота в галузі створення та впровадження практичних криптографічних алгоритмів з використанням еліптичних кривих. Цьому сприяв швидкий розвиток мікропроцесорної техніки, який зажадав реалізації криптографічних алгоритмів на цій техніці, а алгоритми, що існували тоді, явно не підходили для цієї мети. На сьогодні еліптична криптографія досягла такого рівня розвитку, який дозволяє включати еліптичні криптографічні алгоритми до національних та міжнародних стандартів. Вже прийнято стандарти ANSI X9.62 та X9.63 (банківська сфера, 1999 р.), національний стандарт США FIPS 186-2 (2000 р.) та стандарт IEEE P1363-2000. Розглядається включення криптографічних алгоритмів на еліптичних кривих до стандартів ISO/IEC 14888 (цифровий підпис), ISO/IEC 9796 (цифровий підпис із відновленням повідомлення), ISI/IEC 14946 (цифровий підпис, спрямоване шифрування, транспортування ключів та встановлення загального ключа). Відомо, що ведуться роботи з включення алгоритмів еліптичного типу до низки галузевих стандартів, у тому числі стандартів забезпечення безпеки в Інтернеті.

Вся сукупність відомих теоретичних результатів та власний досвід роботи з еліптичними кривими переконливо показують, що еліптичні криві дають унікальні можливості для побудови надійних та ефективних криптографічних алгоритмів. Основні переваги груп, пов'язаних з еліптичних кривих, такі. 1. Відносна простота обчислення параметрів цих груп. 2. Відсутність ефективних алгоритмів розв'язання задачі дискретного логарифмування в цих групах та вкрай мала ймовірність їх появи у майбутньому. Це дозволяє використовувати ключі невеликої довжини із гарантією дуже високої стійкості. Наприклад, еліптична криптосистема з ключем довжиною 160 біт еквівалентна за стійкістю алгоритму стандарту, що діє, а при ключі довжиною 320 біт вона еквівалентна за стійкістю звичайному алгоритму з ключом довжиною 5200 біт. 3. Якщо обчислення параметрів групи (насамперед, самої еліптичної кривої та її порядку) пов'язане з певними технічними складнощами, то обчислення в цих групах дуже ефективні і легко реалізуються в апаратному вигляді. 4. Дуже широка можливість вибору еліптичних кривих та пов'язаних із ними груп при фіксованому базовому полі. У звичайних алгоритмах такого вибору практично немає. 5. Наявність ясних та простих необхідних та достатніх умов, виконання яких виключає застосування відомих специфічних для еліптичних кривих методів криптоаналізу. Тому державний стандарт України розроблено на основі криптографічних перетворень на еліптичні криві. Для забезпечення високої криптографічної стійкості цифрового підпису еліптичні криві та поля їх визначення повинні відповідати жорстким вимогам. Такі вимоги вироблені з урахуванням аналізу відомих методів розв'язання завдання дискретного логарифмування на еліптичних кривих.

1.3 Ефективні алгоритми генерації еліптичних кривих

Створено ефективні обчислювальні алгоритми генерації еліптичних кривих із заданими криптографічними властивостями. Постараємося зробити деякий порівняльний аналіз сучасних алгоритмів обчислення та перевірки ЕЦП, які ґрунтуються на задачі дискретного логарифмування у групі точок еліптичної кривої. В даний час стійкість всіх несиметричних криптографічних перетворень, які практично застосовуються в алгоритмах обчислення цифрового підпису та включені до національних та міжнародних стандартів, заснована на складності розв'язання одного з двох завдань, задачі факторизації та задачі дискретного логарифмування. Спроби використовувати в криптографічних цілях інші обчислювально складні математичні завдання (наприклад, задачу про рюкзак або задачу декодування лінійних кодів загального виду) до успіху не привели. Широке практичне використання таких несиметричних криптографічних систем привернуло увагу дослідників до створення нових способів розв'язання двох фундаментальних задач теорії чисел – задачі дискретного логарифмування та факторизації. У результаті було створено досить потужні алгоритми субекспоненційної складності розв'язання цих завдань, причому складність розв'язання обох завдань однакова, хоча ці завдання не зводяться одне до одного. Криптографічні алгоритми на еліптичних кривих будуються цілком аналогічно звичайним теоретико-числовим алгоритмам. Фактично для будь-якого такого алгоритму можна побудувати еліптичний аналог шляхом заміни основного криптографічного перетворення - зведення в ступінь - операцією скалярного множення групи точок еліптичної кривої. Усі стандарти, що визначають еліптичні криптографічні перетворення, побудовані саме в такий спосіб. Перший чинний стандарт цифрового підпису на основі еліптичної криптографії - стандарт Американського національного інституту стандартів ANSI X 9.62-1998 "Public Key Cryptography for Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)", орієнтований на застосування у банків У цьому стандарті використовуються еліптичні криві над простим кінцевим полем GFp, p – просте число, та кінцевим полем   m GF 2 характеристики 2. Як алгоритм обчислення та перевірки цифрового підпису використовується еліптичний аналог алгоритму DSA (алгоритма Кравиця), визначений у першій редакції національного стандарту цифрового підпису США FIPS 186 (1993 р.) та збереженого в наступних редакціях FIPS 186-1 (1996 р.) та FIPS 186-2 (2000 р.). Під час обчислення цифрового підпису використовується одноразовий секретний ключ k. Секретні ключі цифрового підпису формуються за допомогою датчика випадкових чисел, визначеного стандартом ANSI X 9.17 “American National Standard”. Financial Institution Key Management (Wholesale)”, 1985. Крім того, допускається використання датчика випадкових чисел, наведеного у додатку до стандарту. У цьому додатку повторюються датчики, визначені стандартом FIPS 186-2, в яких як криптографічного перетворення застосовуються алгоритм шифрування DES (ANSI X 9.92 Data Encryption Algorithm) або функція хешування SHA-1 (Національний стандарт США FIPS 180-1 Secure Ha Standard”, 1995 та ANSI

X 9.30 “American National Standard for Financial Services. Public Key Cryptography За допомогою Irreversible Algorithms для Financial Services Industry. Part 2. The Secure Hash Algorithm”, 1993). Вихідне повідомлення перетворюється на рядок довжиною 160 біт за допомогою функції хешування SHA-1, цей рядок перетворюється на ціле число e. Основне криптографічне перетворення виконується шляхом обчислення точки еліптичної кривої Q x1, y1 kG, k - разовий секретний ключ, потім елемент базового поля 1 x перетворюється в ціле число 1 x. Далі підпис обчислюється так само, як і в стандартному алгоритмі DSA. Причина використання алгоритму DSA у цьому стандарті є зрозумілою. Алгоритм DSA використовується давно і добре знайомий західним користувачам. З опису процесу обчислення цифрового підпису видно, що основна маса обчислень з цілими числами за порівняно невеликим модулем залишається без змін, скалярним твором замінено трудомістку операцію - зведення в ступінь за великим модулем. Тому реалізації старого стандарту досить легко переробити у реалізацію нового стандарту. Національний стандарт цифрового підпису США FIPS 186 "Digital Signature Standard (DSS)" був прийнятий в 1993 р. і включав теоретико-числовий алгоритм DSA. У 1995 р. до нього був включений алгоритм обчислення та перевірки цифрового підпису з використанням алгоритму RSA. Це було зроблено шляхом простого включення стандарту ANSI X9.31 “Digital Signatures Using Reversible Public Key Cryptography for Financial Services Industry (rDSA)”. Новий стандарт отримав назву FIPS 186-1. Аналогічно до цього стандарту було включено алгоритм обчислення та перевірки цифрового підпису на еліптичних кривих. До нового стандарту FIPS 186-2 просто включено посилання на описаний вище стандарт

У разі поля характеристики 2 рекомендовано 5 пар еліптичних кривих, для кожного поля одна випадкова крива та одна аномальна. Стандарт IEEE P1363-2000 "Standard Specifications for Public Key Cryptography" є результатом виконання дуже великого проекту, мета якого полягала в нормуванні майже всіх криптографічних алгоритмів, що використовують несиметричні перетворення, заснованих на задачі дискретного логарифмування в мультиплікативної групи дискретного логарифмування групи точок еліптичної кривої. У цей стандарт включено два алгоритми обчислення та перевірки цифрового підпису на еліптичних кривих. Один з них - алгоритм ECDSS, що точно збігається з алгоритмом, включеним до стандарту X 9.62 (і FIPS 186-2). Другий алгоритм – еліптична версія алгоритму обчислення та перевірки цифрового підпису з відновленням повідомлення Нюберга-Рюппеля [4]. Очевидно, що алгоритм Нюберга-Рюппеля суттєво простіше алгоритму DSA, в останньому, наприклад, використовується досить трудомістка операція обігу цілих чисел за модулем простого числа. У теоретико-числовому випадку ця негаразд помітно, основна криптографічна операція виконується з цілими числами, тому повномасштабна арифметика виконання операцій із цими числами потрібна у разі. При переході до еліптичних кривих бажано звести арифметику цілих чисел до найнеобхіднішого мінімуму

Стандарт P 1363-2000 рекомендує використовувати як функцію хешування або функцію SHA-1, або функцію RIPEMD 160. Зараз ведеться робота з включення алгоритму цифрового підпису на еліптичних кривих у стандарти ISO, стандарти Інтернету (RFC) та ряд галузевих стандартів. У 2001 р. у Російській Федерації було прийнято стандарт Р34.10-2001 “Інформаційна технологія. Криптографічний захист інформації. Процеси формування та перевірки цифрового підпису”. Цей стандарт відрізняється від стандарту Р34.10-94 тільки тим, що операція зведення у ступінь у циклічній групі простого кінцевого поля замінена скалярним множенням групи точок еліптичної кривої, визначеної над простим кінцевим полем. Стійкість нового стандарту суттєво підвищилася, проте структурні недоліки, на які ми вказували вище, залишились. Зазначимо також, що використання простого кінцевого поля як визначення еліптичної кривої роблять обчислювальні процедури формування та перевірки підпису досить повільними. Отже, можемо констатувати, що більшість прийнятих і розроблюваних державних та галузевих стандартів, що використовують еліптичну криптографію, зберігає алгоритми, що раніше діють у цих стандартах.

Перехід на особливості еліптичних кривих був зроблений заміною операції зведення в ступінь у циклічній групі простого кінцевого поля на множення на скаляр у групі точок кривої еліптичної, визначеної над простим кінцевим полем. З одного боку використання старих алгоритмів зручно, вони вивчені та звичні. З іншого боку, при такому підході втрачається можливість скористатися значно простішим алгоритмом і, як наслідок, значно прискорити процес обчислення та перевірки підпису. Подаємо таблицю, де діючі сьогодні алгоритми зіставляються за низкою критеріїв (див. табл. 2). Таблиця 2 – Характеристики діючих криптографічних стандартів, що використовують особливості еліптичних кривих Найменування стандарту, область його застосування Використовуваний криптоалгоритм Спосіб переходу на еліптичну криптографію Спосіб завдання еліптичної кривої Можливість використання нефіксованих значень довжини ключа ANSIX9.69 ANS. Заміна зведення в ступінь на скалярне множення в групі точок еліптичної кривої Задано алгоритми Так FIPS 186-2 (2000 р.) Національний стандарт США DSA ˉ"ˉ Наведено список кривих Так IEEE P1362-2000 DSA Нюберга-Рюппеля ˉ" P34.10-2001 Росія Р34.10-94 ˉ"ˉ Відсутня Ні ДСТУ 4145-2002 Україна Нюберга-Рюппеля ˉ"ˉ Наведені рекомендовані криві Так Звідси видно, що ДСТУ 4145-2002 виграє алгоритм Нюберга-Рюппеля і, як наслідок, при всіх інших рівних якостях має вищу швидкість обчислення і про вірки підпису. На закінчення відзначимо, що успіхи математичних досліджень у галузі еліптичної криптографії дали можливість отримати: 1) досить прості алгоритми генерації еліптичних кривих; 2) абсолютно ясні, прості, необхідні та достатні умови, виконання яких дає можливість виділити з безлічі одержуваних еліптичних кривих ті, які мають необхідні властивості та придатні для криптографічних застосувань. 3) достатні гарантії того, що завдання дискретного логарифмування в полі точок еліптичної кривої залишатиметься у межах експоненційної складності.

2 ДОСЛІДЖЕННЯ МОЖЛИВОСТЕЙ КРИПТОАНАЛІЗУ

2.1 Аналіз існуючих підходів до факторизації

Розглянемо традиційний підхід до факторизації багаторозрядних чисел.

Весь набір існуючих методів не вирішує завдання факторизації в принципі, так як майже всі решіткові та інші алгоритми побудовані на жорсткому зв'язку та залежності часу їх виконання від розрядності числа *N*, що факторизується. Але зауважимо, що у чисел є й інші властивості крім розрядності, які можна використовувати у алгоритмах факторизації.

Оцінки складності – евристичні. Найбільш вживаним алгоритмом факторизації є алгоритм, що використовує теорему Ферма. Це проста ідея факторизації цілого непарного числа *N* історично полягає в пошуку пари квадратів чисел різної парності, різниця яких кратна *kN*, при *k* =1 розкладання успішно реалізується так як в цьому випадку відразу отримуємо добуток двох многочленів  з співмножниками *N*. При *k*>1 трапляються тривіальні розкладання.

Таким чином, проблема факторизації перетворюється на проблему пошуку відповідних квадратів чисел. Розуміли ці факти багато математиків, але П. Ферма першим в 1643 реалізував ідею пошуку таких квадратів в алгоритмі, названому його ім'ям. Перепишемо інакше наведене співвідношення .

Якщо різниця ліворуч від рівності не дорівнює квадрату, то змінюючи *х*, можна підібрати інший квадрат, щоб і праворуч виходив квадрат. Практично всі нинішні алгоритми використовують цю ідею (пошуку пари квадратів), проте навіть не виконуючи обчислень, лише накопичувавши суму для досягнення развязку потрібно занадто багато часу.

Криптографія з відкритим ключем (асиметричні криптосистеми) нині стала вже повсякденністю, хоча мала зовсім невеликий часовий період розвитку(в історичному плані). До основних її відмінностей від симетричної криптографії слід насамперед віднести можливість використання передачі (шифрованих текстів (ШТ), інших повідомлень) незахищених каналів у мережах інформаційного обміну, цифровий підпис, аутентифікацію.

Математичною основою цих фактів та явищ є:

- одностороння (односпрямована) функція;

- Одностороння функція з лазівкою (з секретом).

Функція *f: X* → *Y* називається односторонньою, якщо існує ефективний алгоритм її обчислення за будь-якого *Х*, але не існує такого алгоритму для обчислення зворотної до неї функції.

Функція *f: X → Y* називається односторонньою із секретом (з лазівкою, потайним ходом, trapdoor), якщо за наявності деякої додаткової інформації (ключа) можливий ефективний алгоритм її поводження. Під ефективним алгоритмом розуміють поліноміальний на відміну експоненціального.

Математикам давно знайома проблема факторизації, і вони шукають шляхи її ефективного вирішення. До основних досягнень (результатів) математики в галузі факторизації чисел слід віднести наступне:

— метод Евкліда відшукання найбільшого загального дільника (НСД), у якому пари чисел *b* і *а*, якщо вони складові, але взаємно прості, НСД (*а*, *b*)=1. Але якщо ці числа мають спільний дільник, то алгоритм НСД знаходить найбільший дільник *d* і НСД (*а*, *b*)=*d*.

- Інший важливий результат, який має математика, полягає в тому, що будь-яке складове непарне ціле число N представимо різницею квадратів цілих чисел різної парності  або інакше . Друге співвідношення припускає  ціле. Ці результати поєднуються в одному співвідношенні .

- Метод факторизації Ферма відомий з 1643 року. Він знаходить найбільший множник *d* числа *N*, що не перевищує . Метод вимагає  арифметичних операцій і найшвидше, якщо множники *p* і *q* мають близькі значення, тобто. коли їхня різниця мала.

- Решето Ератосфена.

### Метод лінійної решітки.

Ріхард Шрёппель (Richard Schroeppel), займаючись проблемою факторизації чисел, запропонував оригінальний алгоритм для генерації співвідношень наступного виду .

На числовому інтервалі , де 0 < e ≤1/2 – фіксоване дійсне число, розглядаються дві функції двох цілих змінних , визначених у цьому інтервалі  рівностями:





де , наприклад, 

В силу порівнянності цих функцій по модулю *N* їх можна використовувати при побудові необхідних співвідношень. Фіксується межа > 0 для елементів факторної бази , Яку утворюють всі прості числа, що не перевищують, а також ціле число -1. Далі розглядається порівняння за модулем *N* .

Припустимо, що права частина порівняння може бути представлена добутклм і ліва частина буде повним квадратом, якщо величини входять до добутку парне число разів , де рi прості числа з факторної бази S, а Δ – ціле число, що є повним квадратом.

У такій ситуації порівняння з творами за модулем *N* відповідає порівнянню  з відомим розкладанням на множники з факторної бази. Метою в методі є отримання таких співвідношень числом, що перевищує кількість елементів факторної бази.

Якщо це досягається, то можна побудувати, використовуючи алгоритм гауссового виключення, порівняння , що забезпечує розкладання *N* на множники.

Для формування величин (*a, b*) існує ефективний спосіб, званий решетом, пошуку значень , при яких величини (*a, b*) розкладаються у добуток елементів факторної бази.

І самі величини (*a, b*) набувають невеликих значень. Значення обмежені та вірна оцінка:  Перевірка ділимості величини s (a, b) на просте число р для довільних a, b∈ I зводиться до перевірки подільності на величини . Це легко показується. Нехай просте число *р* ∈ *S* поділяє значення величини *s* (*a, b*) при деяких *а*. Але з рівності:







слідує, те , що з довільних цілих значень *k*, l.

Істотним недоліком методу є необхідність великого обсягу пам'яті, і відсутність алгоритмічного способу побудови порівнянь з відомим розкладанням лівої частини у добутку множників з факторної бази. Цей текст (компіляція) запозичений із різних публікацій, але практично не супроводжується моїми коментарями. Для порівняння наводжу інший текст про метод QS, але супроводжую коментарями, які, як я сподіваюся, вносять розуміння та ясність.

### Квадратичне решето.

Цей алгоритм ( quadratic sieve- QS) запропонований у 1981 році Карлом Померансом і майже 10 років був кращим (до запропонованого в 1990 алгоритму SNFS (special number field sieve), що забезпечив розкладання на множники 9-го числа П.Ферма (155 десяткових знаків) ).

Можливості квадратичного решета обмежуються розкладанням у співмножники чисел, з не більше ніж 110 десяткових цифр у їх описі. Оцінка складності (евристична) алгоритму навіть після численних удосконалень складає  арифметичні операції.

При розгляді QS важливими поняттями, які бажано утримувати в голові, є:

- вектор показників ступенів чисел -<4 0 1 1>, де, наприклад, для маємо 

- лінійна залежність векторів;

- факторна база S = (2,3,5,7) набір малих простих чисел, для яких N -квадратическое відрахування;

- гладкі числа відрахування, які <N і розкладається в прості числа з бази S;

- процедура просіювання.

Метою методуQS є факторизація натурального числа N шляхом отримання для чисел *х* і *у*, множина значень які будуються спеціальним чином, співвідношення  і потім перевірка за допомогою НСД співвідношення , що завершує факторизацію *N*.

Дільники p і q знаходяться за допомогою спеціальних чисел-значень многочлена виду , де  і  пробігає значення в діапазоні -*М*<*j*<*М*. Значення *q*(*x*) у цілих точках таблиці 1.1 є квадратами (mod *N*).

Очевидно, що збільшення значення *М* призведе до збільшення множини чисел х для просіювання, що сприяє зростанню множини гладких чисел. Але при цьому зростає і час просіювання. Аналогічні проблеми виникають і при збільшенні факторної бази *S* більше елементів в *S* довше йде обробка таблиць і Q (в останній опущені аргументи *х*).

Факторна база *S* — набір малих простих чисел, що включає *р* =-1 і прості числа *р*i, *р*i≤*В*, такі, що символ Лежандра (*N/pi*)=+1, тобто *N* є квадратичним залишком для всіх *рi* з факторної бази *S*.

Пара цілих чисел  називається гладкою щодо факторної бази *S*, якщо: виконується порівняння  і *b* розкладається у добуток елементів *S*.

У методі QS формується безліч аргументів *х* і чисел *q*(*x*), серед яких відшукуються гладкі числа відносно факторної бази. Ці числа використовуються для побудови співвідношень, декларованих з метою методу. Нижче на прикладі наводиться варіант такого формування (таблиця 1.1).

Після формування послідовності пар значень (*х*, *q*(*x*)) виконуємо просіювання і знаходимо значення *хi*, для яких , тобто *q*(*x*) розкладається на співмножники у нашій факторній базі *S*.

Якщо процес формування послідовності пар значень (*х*, *q* (*x*)), процес просіювання під час читання тексту вимагає нашої уваги значно більше. Нехай заданий багаточлен *q*(*x*) є *Z*[*x*]. Будемо відшукувати всі цілі числа *K* у відрізку [-*М*, *М*] такі, що деякі значення *q*(*x*) розкладаються у добуток простих чисел із заданої множини *S*.

Ідея алгоритму просіювання множини чисел полягає в тому, що якщо , то за будь-якого цілому *ℓ* виконується також порівняння з нулем для . Це забезпечує фіксування одразу деякої множини чисел х, для яких у розкладанні чисел *q*(*x*) на прості співмножники входить просте число *р* у ступені не меншої *t*. Це спостереження значно прискорює процес пошуку (просіювання) необхідних гладких чисел.

Усі теоретичні положення QS зручно оформити як кроків алгоритму:

1. Вибираються границі *P* і *A* порядку  вид багаточлена для просіювання  та діапазон значень.

2. Для аргументу , … і многочлена *q*(*x*) виписуємо до таблиці по порядку пари цілих чисел 

3. Для кожного непарного простого числа *p*≤*P* перевіряємо умову Лежандра (*N*/*p*)=1 і, якщо вона не виконується, видаляємо *p* з факторної бази. Однак, якщо ми розкладемо кожне значення *q*(*x*) за ступенями простих чисел, то побачимо, що деякі числа з них не мають множників більше за поріг (у прикладі в розкладанні у 4-х чисел всі фактори *p*≤11), а *р* = 5 не є дільником жодного з них, отже, факторну базу прикладу ми обмежимо такими простими числами *S* = (2,3,7,11):

.

4. Припускаючи, що p таке непарне просте число, що *N*-квадратическое відрахування по модулю *p* обчислюємо рівність  для 

Беремо значення *β* у порядку зростання, доки виявиться, що рівняння немає рішень *x*, обчислених по модулю  з якимось із чисел у області . Позначимо через *β* найбільше з таких чисел, для яких у зазначеній ділянці знайдеться число *x* із властивістю .

Нехай - два розв’язки   і . Не потрібно, щоб  належали відрізку  

5. При тому ж значенні *p* переглядаємо список значень , отриманий у пункті 2. У стовпці, що відповідає *p*, ставимо 1 проти всіх значень , для яких x відрізняється від *x*1 на деяке кратне *p*. Після цього замінюємо 1 на 2 для всіх таких значень , що *x* відрізняється від *x*1 на кратне .

Потім замінюємо 2 на 3 у всіх значень , для яких x відрізняється від *x*1 на кратне , і так далі . Потім робимо те саме з замість . Найбільшим числом, яке з'являється у цьому стовпці, буде *β*.

6. Щоразу, коли в пункті 5 ставимо 1 або замінюємо 1 на 2, 2 на 3 і т.д., ділимо відповідне число на p і зберігаємо отриманий результат у тій самій комірці.

7. У стовпці під *p* = 2 при *N*≢ 1(mod 8) просто ставимо 1 проти  з непарним x і ділимо відповідне на 2. При N≡ 1(mod 8) вирішуємо рівняння і продовжуємо в точності так само, як у разі непарного p (за винятком того, що при *β* ≥ 3 рівняння буде мати 4 різні рішення 

8. Коли вказані дії будуть проведені для всіх простих чисел, що не перевищують *P*, відкинемо всі , Крім тих, які звернулися в 1 після поділу на всі ступеня *p*, не перевершують *P*.

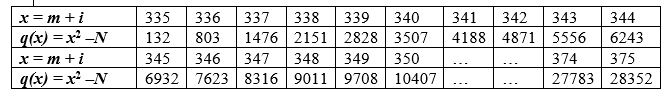
Тоді вийде таблиця того ж виду, що в прикладі методу факторизації Ферма, в якій стовпець *b*i буде містити всі такі значення *x* з інтервалу , що  є *B*- число, а інші стовпці відповідатимуть тим значенням *p* ≤ *P*, для яких *N* – квадратичний залишок.

9. Частина процедури, що залишилася, в точності збігається з процедурою з факторизації Ферма.

Застосовуватимемо послідовно положення теорії алгоритму. Нехай встановлено складове число N = pq = 112093, яке потрібно розкласти на множники. Наслідуючи концепцію квадратичного решета, витягуємо квадратний корінь з N,  і округляємо його значення до більшого цілого.

Далі в методі необхідно сформувати для *i* = 1 (1) А таблицю пар чисел , що зростають з кроком 1 значень аргументів (обмежимося 40 значеннями) та відповідних їм значень функції.

Таблиця 2.1 - Множина, що містить *В*-гладкі числа (значення *q* (*x*)) з цілими аргументами.



У цій таблиці бажано мати гладкі числа. Їх треба виявити та залишити для подальшої обробки, а решту можна просто видалити. Основна ознака для видалення деякого числа - наявність серед його дільників великих множників (хоча б одного), які не увійшли до факторної бази. Як межа гладкості приймемо *В* = 11.

Є можливість у процесі рішення вибране безліч чисел змінювати (зменшувати, збільшувати). У таблиці серед значень *q*(*х*) немає повних квадратів. Тепер можна приступити до пошуку чисел, які забезпечать отримання однорідної системи лінійних рівнянь за модулем 2. Сама ця система нетривіальними рішеннями матиме змінні, добутки яких отримуватимуть значення, що дорівнюють повному квадрату.

### Просіювання.

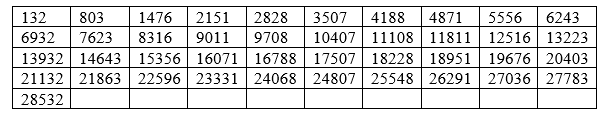
Розглянемо у деталях як із множин значень  у таблиці виявляються *В*-гладкі числа, без представлення кожного їх добутком простих співмножників.

Вже сформована 2-ма рядкова таблиця пар чисел (*х*-верхній рядок, *q*(*x*) — нижній). Усі підряд значення *q*(*x*) (нижнього рядка таблиці ) послідовно ділять на всі *р*j — прості числа (та їх ступеня) з факторної бази *S*. Результати розподілу вписують у попередні позиції.

Ті значення, які діляться на *р*j залишають без змін. Мета таких перетворень – не викреслити числа, що повністю розділилися, з таблиці, а отримати в результаті поділу в їх осередках поодинокі значення, що відповідає В-гладким числам, тобто. повному розкладанню числа таких позиціях на ступеня простих з *S*. Також серед цих значень виявляється аргумент *х*.

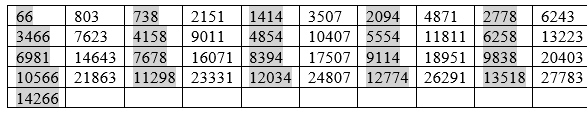
Далі оброблятимемо (операція ділення) числа з рядка таблиці   таблиці, які зберемо в одній новій ***Q*** таблиці (41 позиція). Усі числа цієї нової таблиці ділимо на *р* = 2, *β* = 1. Аргументи в таблицю не пишемо, оскільки це просто номери значень.

Таблиця 2.2 – Результати з рядка 



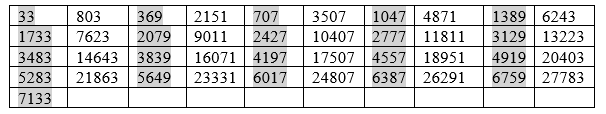
В результаті отримуємо всі парні числа, поділені навпіл у таблиці *Q* , стовпці парних чисел чергуються з непарними. Це є закономірним. У вихідній таблиці парні чергувалися з непарними числами. Для будь-якого цілого *d* виконується рівність , так що якщо число з набору *q*(*х*) ділиться на *d*, то всі числа віддалені від нього на відстані, кратному *d* також діляться на *d.*

Таблиця 2.3 – Таблиця *Q*



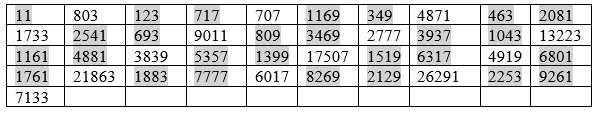
Числа, що змінилися, виділені заливкою в осередках, ділимо на (*р* = 2 *β*=2), тобто. ще раз.

Таблиця 2.3 – Таблиця *Q.* Результат ділення на 2



Тепер усі числа у таблиці *Q* непарні. Розподіл на простий *р* =2 та його ступеня 2×2=4 завершилося. Починаємо ділити на таке просте число з факторної бази (*р* = 3, *β* = 1).

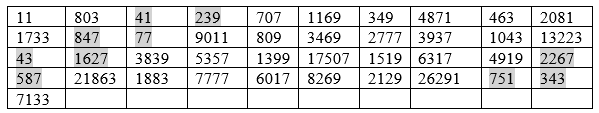
Таблиця 2.4 – Таблиця *Q.* Результат ділення на просте число.



Якщо деяке число розділилося на три, кожне третє після нього також ділиться на три. Таких ланцюжків чисел у таблиці *Q* виникло два: 1-ий ланцюжок починається з першого осередку; 2-ий - з третього.

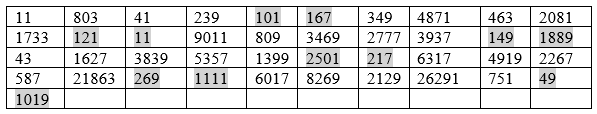
Це наслідок того, що квадратичне порівняння за модулем *р* має два рішення (якщо взагалі має рішення). Ділимо на трійку всього три рази (*р* = 3, *β* = 3), поки таблиця при розподілі на неї не перестане змінюватися.

Таблиця 2.5 – Таблиця *Q.* Результат ділення на 3



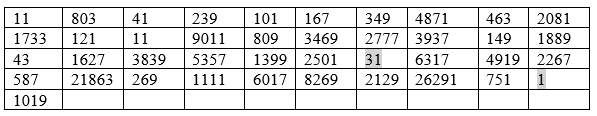
Наступний етап ділення на 3 таблицю *Q* не змінює. Переходимо до поділу на (*р* = 7, *β* = 1) тобто. до наступного факторної бази. Знову виникли 2 ланцюжки з початком у 4-му та в 5-му осередках.

Таблиця 2.6 – Таблиця *Q.* Етап ділення на 7



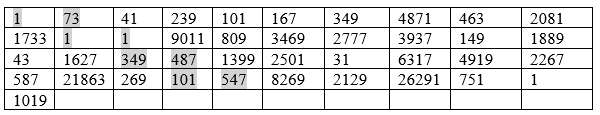
Ще два проходи з дільником (*р* = 7, *β* = 3) призводять до появи першої одиниці таблиці (передостаннє число таблиці 49:7:7 =1).

Таблиця 2.7 – Таблиця *Q.* Етап ділення на 7



Подальше розподіл на 7 таблицю не змінює. Переходимо до поділу на (р = 11, β = 2) число з факторної бази, причому ділимо дворазово і в результаті отримуємо останню таблицю просіювання.

Таблиця 2.7 – Результуюча таблиця



У осередках таблиці (з аргументами х=335, 346, 347, 374) з'явилися нові одиниці. У позиціях останньої таблиці, що містять одиниці у вихідному заповненні, вже розміщувалися 11-гладких чисел, але ми дізналися про це тільки після просіювання. Ось ці числа: 132;7623;8316;27783.

Жодне із значень таблиці виду  не квадрат. Але нам взагалі потрібні квадрати, отже, треба підібрати такі підмножини з *В*-гладких чисел (у таблиці *Q* їм відповідають в осередках одиниці), які при їх перемноженні забезпечують формування повних квадратів.

Вище показано, як ці чотири *В*-гладкі числа знайдені, без розкладання всього списку чисел на множники. Відповідна система лінійних рівнянь A×X = 0(mod 2) має матрицю коефіцієнтів за модулем 2 наступного виду:

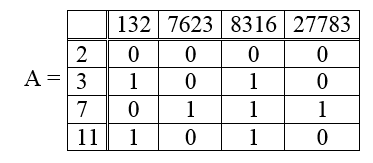


Рисунок 2.1 - Матриця коефіцієнтів за модулем 2

Верхній рядок матриці - гладкі числа, лівий стовпець-факторна основа. Наступним кроком нами отримана нетривіальна лінійна комбінація за модулем 2, в якій коефіцієнти 0 і 1, що дає в результаті нульовий вектор. Для цього отримані результати (*В*-гладкі числа) використовують для формування системи лінійних алгебраїчних рівнянь (СЛАР), АХ≡0(mod2) розв’язок якої призводить до остаточного результату-факторизації *N*.

Формується *А* матриця СЛАР, в якій вектори показників гладких чисел записуються в стовпцях, а вектор *Х* - невідомий. Рішенням цієї СЛАР за модулем 2 виходить вектор . При  — це рішення нетривіальне: (1110), що означає — повним квадратом буде добуток перших трьох чисел.

.

У попередніх позначеннях отримані залежності *X* = П *x*i = 335 · 346 · 347 = 40220770, y = 91476. Тут *X* - *y* = 40220770 - 91476 = 40129294. Знаходимо НСД (*X* - *y*, *N*) = НСД (40129294, 112093) = 112093.

На жаль, знайдений дільник числа *N* = 112093 тривіальний. Зауважимо, що і НСД (*X* + *y*, *N*) = НСД (40312246, 112093) = 1 - тривіальний дільник.

Візьмемо тоді інше рішення системи *АХ*≡0(mod 2) , а саме рішення (0101), що виходить при . Воно означає, що точним квадратом буде твір другого і четвертого чисел:

.

Маємо *X* = Пxi = 346 · 374 = 129404, y = 14553. Знаходимо НCД (*X* - *y*, *N*) = НСД (114851, 112093) = 197. Знайдений нетривіальний дільник 197 числа N = 112093. Тепер число N можна розкласти на множники: N = 112093 = 569 · 197.

## Метод решета числового поля.

Найшвидший на сьогоднішній день метод факторизації натуральних чисел (10 років – одне число). Відмінність цього методу решета числового поля (як спеціального Special Number Field Sieve SNFS) так і загального (General Number Field Sieve - GNFS) від алгоритму QS - квадратичного решета, перш за все, полягає в різних процедурах просіювання, яка проводиться в GNFS не в кільці цілих чисел *Z*, а алгебраїчному числовому полі, що відбилося у назві методу.

Як і методі квадратичного решета в основу формування безлічі чисел для просіювання покладено многочлен не 2-ї, а довільної ступеня *d*>2, у точці *x=m* що задовольняє умові . Факторна база набула іншого вигляду і складається з нерозкладних простих елементів кільця цілих чисел алгебри. І головне — у роботі присутні числові приклади, що супроводжують текст.

Елемент *θ* називається алгебраїчним над полем, якщо він є коренем якогось багаточлена *f*(*x*) з коефіцієнтами поля *К*.

Багаточлен найменшого ступеня зі старшим коефіцієнтом 1, що має *θ* своїм коренем, називається мінімальним многочленом для *θ*. Степенем алгебраїчного елемента *θ* називається степінь його мінімального багаточлена. Числами, пов'язаними з *θ*, називаються решта всіх чисел корінь цього мінімального многочлена.

Перевірка ділимості під час просіювання виконується з використанням норми алгебраїчного числа. Як і методі QS цей спосіб відшукує гладкі числа порядку кореня квадратного з *N*. Розмір чисел зі зростанням *N* експоненційно зростає. Зростання ефективності методу забезпечується тим, що самі числа в числовому полі менші, ніж у кільці, і ймовірність для них стати гладкими трохи вища ніж у методі QS, але самі алгоритми в NFS істотно складніші.

Приміром, автор [3] вважає неправомірним називати цей метод алгоритмом, оскільки у ньому різних етапах використовується кілька окремих самостійних алгоритмів. Інші автори недостатньо підтримують таку позицію.

Метод GNFS включає кілька відправних положень, а також перераховані нижче дії з обробки математичних об'єктів та структур.

1. Нехай задано N – велике складне непарне число, яке потрібно факторизувати. Виберемо неприведений многочлен, ступінь якого *d*>3 (при *d*=2 нічого очікувати виграшу проти методом квадратичного решета).

2. Виберемо ціле m таке, що , і розкладемо *N* на підставі *m*:

. (1)

3. Зв'яжемо з розкладанням (1) неприводимый багаточлен від х в кільці Z[x] багаточленів з цілими коефіцієнтами . У точці m 

4. Визначимо багаточлен просіювання  як однорідний багаточлен від двох змінних:

 (2)

5. Визначимо другий багаточлен  і відповідний однорідний багаточлен просіювання з іншої факторної бази 

6. Виберемо два позитивні числа , що визначають область просіювання (англ. sieve region): SR =  у формі прямокутника.

7. Нехай *θ* - корінь . Розглянемо кільце поліномів Z [θ]. Визначимо безліч кільця, яке називається алгебраїчною факторною базою FB1, що складається з багаточленів першого порядку виду з нормою (2), що є простим числом. Ці многочлени - прості нерозкладні в кільці цілих алгебраїчних цілих поля *K = Q* [θ]. Обмежимо абсолютні значення норм поліномів із FB1 константою B1.

8. Визначимо раціональну факторну базу FB2, що складається з усіх простих чисел, обмежених зверху константою B2.

9. Визначимо множину FB3, яка називається факторною базою квадратичних характерів. Це множина поліномів першого порядку норма яких - просте число. Повинна виконуватись умова FB1∩FB3 =∅.

10. Виконаємо просіювання багаточленів

 по факторній базі FB1 і цілих чисел

 по факторній базі FB2. В результаті отримаємо множину M, (табл. Г) що складається з гладких пар (a, b), тобто таких пар (a, b), що НСД (a, b) = 1, багаточлен і число  і  повністю розкладаються по базах FB1 та FB2 відповідно.

11. Знайдемо (як і в методі QS) таке підмножина S ⊆M, що

; тут Nr - позначення норми многочлена;

.

12. Визначимо багаточлен ,

де  — похідна .

13. Багаточлен g(θ) є повним квадратом у кільці багаточленів Z(θ). Нехай тоді α(θ) є квадратний корінь із g(θ) і B — квадратний корінь із .

14. Будуємо відображення ф: θ → m, замінюючи поліном α(θ) числом α(m). Це відображення є кільцевим гомоморфізмом кільця цілих алгебраїчних чисел  в кільце Z, звідки отримуємо співвідношення:



.

15. Нехай B = f '(m) · C. Знайдемо пару чисел (A, B) таких, що Тоді знайдемо дільник числа N, обчислюючи НОД(N, A ± B), як це робиться, наприклад, методом квадратичного решета. Це множина пунктів можна укрупнити до трьох етапів. Як і методі QS спочатку формується масив вихідних значень, серед яких очікуються гладкі, факторна база з оборотних елементів Z/nZ, що включає елементи , де *р* збігає до деякої кінцевої множини індексів . Символом позначимо вектори .

Далі розглядається відображення  .

.

2.2 Криптоаналіз асиметричних алгоритмів шифрування

Криптоаналіз - це наука, що вивчає методи злому криптосистем, що дозволяє оцінити якість криптосистем. Криптоаналіз сучасних асиметричних криптосистем є важкоздійсненним завданням, що витрачає катастрофічно велику кількість часу. Тому пошук нових та ефективних методів криптоаналізу є актуальним завданням. Безпека шифру залежить від вибору функції пастки. Функція пастка - це функція, яка дозволяє ефективно обчислити значення, але не дозволяє легко звернути обчислення без кількості підказки. Приклад функції реалізовано алгоритмі асиметричного шифрування RSA. Базовий принцип алгоритму RSA будується на тому, що можна знайти такі числа e, d і n, щоб зведення цілого числа, що шифрується, m в ступінь e · d по модулю n для всіх цілих чисел m дорівнювали m:



Експонент d є закритим ключем і обчислити його вкрай складно, навіть якщо n, e і m відомі. Експонента e та число n у парі є відкритим ключем. Відкритий ключ n складається з добутку двох найпростіших чисел *p* і *q* великого розміру, які відрізняються один від одного на кілька символів для ускладнення криптоаналізу. Якщо *p* і *q* величезні, то процес розкладання добутку *p* · *q* є важкою задачею. Числа *p* та *q* використовуються для обчислення закритого ключа *d*. Розкладання чисел можна здійснити з використанням методу "Квадратичного решета" або "*Ро*-алгоритма Полларда". Для факторизації числа *n* у роботі розглянемо природні алгоритми криптоаналізу: алгоритм мурашиних колоній та алгоритм бджолиної колонії.

Алгоритм мурашиних колоній є імовірнісним підходом до вирішення завдань, що наслідує поведінку мурах. Алгоритм полягає в наступному: дано граф  мурах. Мурахи розміщуються на вершинах графа *G* і будують маршрути. Мурахи можуть бути розміщені випадковим чином. Під час побудови маршруту мурахи відкладають феромон на пройденому шляху. При виборі наступної вершини для переходу враховується рівень феромону, відкладеного на переході з поточної вершини в наступну. Імовірність переходу мурахи на доступні йому вершини визначається такою формулою:



де  - кількість феромону, відкладеного мурахою на ребрі *xy*, *η xy* - привабливість маршруту, *α*, *β* - фактори, які контролюють вплив параметрів  і  ,  - множини ребер, доступні мурашки *k* з вершини *x*. Після побудови маршрутів значення феромонів на ребрах оновлюються за такою формулою:



де - кількість феромону на ребрі xy мурахою *k*, *ρ* - коефіцієнт випаровування,  - кількість феромону, відкладеного мурахою на ребрі xy під час поточної ітерації. Кількість відкладеного феромону  визначається за такою формулою:



де  - ціна переходу по ребру *xy*, *Q* - константа. Якщо умови зупинення алгоритму не виконано, процедура повторюється. Умовами зупинки можуть бути обмеження часу або досягнення оптимального рішення. Для знаходження простого дільника складового числа *N*, приймемо такі параметри алгоритму:



де  - маршрут, побудований мурашкою - функція, що отримує дробову частину від поділу на *N*. *Q* визначає зменшення рівня феромону зі збільшенням довжини маршруту, можна прийняти рівним довжина маршруту. Метою алгоритму є маршрут, .

Алгоритм бджолиної колоній - стохастичний біонічний алгоритм оптимізації, що наслідує поведінку бджіл. У цьому алгоритмі виділяються кілька етапів: початкова розвідка та локальна розвідка. Під час початкової розвідки вибираються C точок. Вибір точок може відбуватися випадковим чином. Далі на вибраних точках відбувається локальна розвідка, де відбувається пошук оптимальнішого рішення в радіусі R на вибраних точках. При поліпшенні значення значення цільової функції F і набір аргументів X зберігаються. Для розкладання складового числа, цільовою функцією приймемо:



Як радіус пошуку приймемо значення R = c/1.442685, де c — кількість цифр у двійковому записі шуканого числа.

2.3 Розробка системи криптоаналізу для асиметричних криптосистем

Система реалізована в 2 модулі: модуль реалізує алгоритм та графічний інтерфейс. Такий спосіб реалізації зручний тому що дозволять легко відокремити логіку реалізації графічного інтерфейсу від алгоритму криптоаналізу. При розробці використовувалися 2 мови програмування: C та C++. C був обраний за рахунок популярності та простоти взаємодії з низьким рівнем програмування, внаслідок чого можна просто оптимізувати програму на швидкість. C++ вибрано залежно від бібліотеки Qt.

Система криптоаналізу складається з наступних компонентів:

* Криптографічний модуль libsodium;
* Бібліотека, що реалізує алгебру з великими числами GMP;
* Графічний інтерфейс.

Модуль криптоаналізу реалізує логіку криптоаналізу та взаємодіє з криптографічним модулем для перевірки проміжних результатів. Дозволяє розкласти складову кількість і цим відновити закритий ключ, звітує про проведені дії. Модуль реалізований мовою програмування C та використовує libsodium та GNU MP.

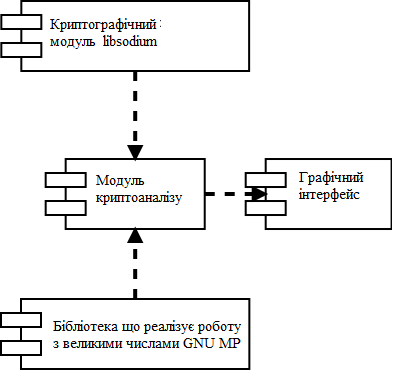


Рисунок 2.2 Структурна схема системи криптоаналізу

libsodium надає низку криптографічних функцій. Використовується для створення якісних випадкових чисел. Бібліотека написана мовою C та поширюється під ліцензією BSD.

GNU MP реалізує роботу з великими числами та надає ряд математичних функцій, включаючи функції теорій чисел. Бібліотека написана з використанням мов C та C++ та поширюється під ліцензіями LGPL та GPL.

Графічний інтерфейс використовується для контролю модуля криптоаналізу та відображення результатів та його роботи. Дозволяє налаштувати модуль криптоаналізу, запустити процес криптоаналізу та отримати результат криптоаналізу. Цей модуль написаний на C++ і використовує бібліотеку Qt5.

Особливості системи:

* Використовує оптимізовані алгоритми (дякую GNU MP).
* Використовує евристичні методи розкладання складового числа.
* Виробляє криптоаналіз у кілька потоків.

Для перевірки ефективності системи було проведено ряд експериментів: порівняння часу роботи алгоритмів від розміру ключа та час роботи алгоритмів залежно від параметрів. Кожен досвід був запущений 10 разів і як результат взято середній час виконання, оскільки алгоритми використовують випадкові числа для роботи і час виконання кожного разу може відрізнятися. Для першого експерименту використовуються такі параметри:

Для алгоритму мурашиних колоній:

* + Кількість мурах: 4
  + Довжина колії: 64
  + Коефіцієнт випаровування феромону: 0.4
* Для алгоритму бджолиної колонії:
  + Кількість бджіл: 4
  + Кількість точок для дослідження: 4096

Результати експерименту наведено на рсиунку 2.3.

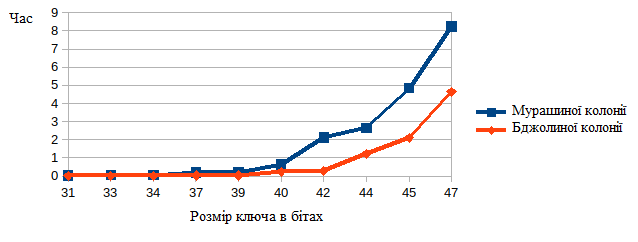


Рисунок 2.3 - Результати порівняльного експерименту

За графіком видно, що алгоритм бджолиної колонії справляється краще із завданням криптоаналізу, ніж алгоритм мурашиних колоній. Експериментально були перевірені лише ключі невеликого розміру, які не є криптостійкими. Це з обмеженням обчислювальних ресурсів устрою у якому проводилися експерименти. Далі проведемо експеримент, який показує вплив збільшення кількості потоків на швидкість алгоритму. Для проведення експерименту були задані такі постійні параметри:

* Розкладне число: 664543753253
* Діапазон пошуку: [903, 815196]
* Для алгоритму мурашиних колоній:
  + Кількість мурах: 4
  + Довжина колії: 64
  + Коефіцієнт випаровування феромону: 0.4
* Для алгоритму бджолиної колонії:
  + Кількість бджіл: 4
  + Кількість точок для дослідження: 4096

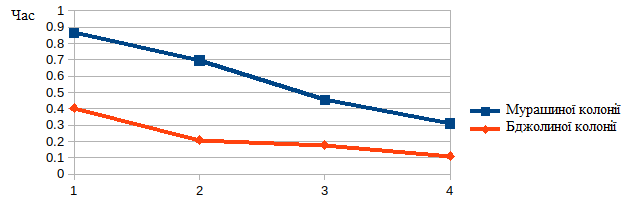


Рисунок 2.4 - Результати порівняльного експерименту з параметром кількості потоків, що варіюється.

За графіком видно, що витрачене на факторизацію числа час лінійно зменшується із збільшенням кількості потоків. В алгоритмі бджолиної колонії розглянемо вплив кількості точок, що досліджуються, на швидкість розкладання числа. Для цього експерименту було прийнято такі постійні параметри:

* Розкладне число: 664543753253
* Діапазон пошуку: [903, 815196]
* Кількість бджіл: 4

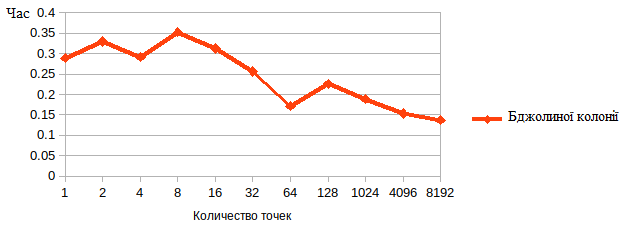
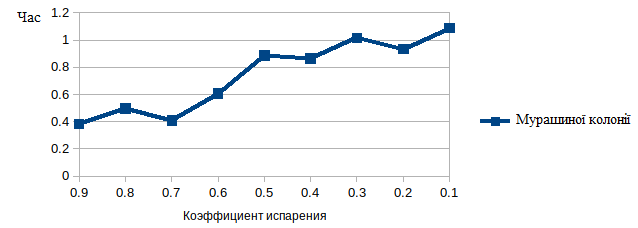


Рисунок 2.5 - Результати експерименту з параметром, що варіюється

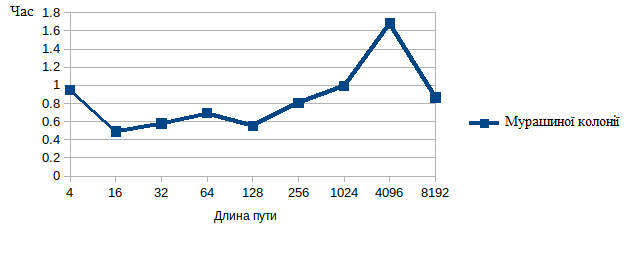
За графіком видно, збільшення кількості досліджуваних точок у середньому зменшує час, затрачуване перебування простого делителя. В алгоритмі мурашиних колоній вплив коефіцієнта випаровування феромону. Для цього експерименту було використано такі постійні параметри:

* Розкладне число: 664543753253
* Діапазон пошуку: [903, 815196]
* Кількість мурах: 4
* Довжина колії: 64

Рисунок 2.6 - Результати експерименту з параметром, що варіюється.

За графіком видно, збільшення коефіцієнта випаровування призводить до збільшення часу на факторизацию. У цьому алгоритмі також розглянемо вплив довжини маршруту швидкість розкладання. Були використані такі постійні параметри:

* Розкладне число: 664543753253
* Діапазон пошуку: [903, 815196]
* Кількість мурах: 4
* Коефіцієнт випаровування феромону: 0.4

Рисунок 2.7 - Результати експерименту з параметром, що варіюється.

За графіком видно, що оптимальне значення довжини маршруту є від 16 до 128. Маршрути коротше або довше призводять до збільшених тимчасових витрат.

У цій статті наводиться порівняльний аналіз п'яти широко відомих асиметричних криптоалгоритмів на основі актуальних критеріїв – показників якості. У ході дослідження з'ясовуються переваги і недоліки криптоалгоритмів, що розглядаються, а також недоліки асиметричної системи шифрування в цілому.

3 ФАКТОРИЗАЦІЯ НА ОСНОВІ ЕЛІПТИЧНИХ КРИВИХ

3.1 Аналіз можливостей криптографії на основі еліптичних кривих

Нинішня інформаційна епоха, що дала людству інструментарій для спрощення процесу спілкування і роботи, також стала причиною появи безлічі актуальних на сьогоднішній день проблем. Забезпечення цілісності та належного рівня конфіденційності інформації, що передається за допомогою мережі Інтернет, є однією з найважливіших проблем, і для її вирішення були розроблені різні симетричні та асиметричні криптоалгоритми. У цій статті мова піде про асиметричні криптоалгоритми. В результаті аналізу сфери захисту інформації було виявлено такі представники асиметричної криптографії:

* RSA;
* DSA;
* шифросистема Ель-Гамалі;
* обмін ключами Діффі-Хелмана;
* протокол Аншеля-Гольдфельда.

Основна мета даної роботи полягає у процесі дослідження перерахованих криптоалгоритмів та їх експертної оцінки, побудованої за заданими критеріями якості.

Таблиця 3.1 - Оцінки, побудованої за заданими критеріями моделі

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 | A5 | Середнє геометричне | Ваги критеріїв |
| A1 | 1 | 3/1 | 7/1 | 5/1 | 5/1 | 3,5 | 0,49 |
| A2 | 1/3 | 1 | 5/1 | 5/1 | 5/1 | 2,11 | 0,29 |
| A3 | 1/7 | 1/5 | 1 | 3/1 | 3/1 | 0,76 | 0,11 |
| A4 | 1/5 | 1/5 | 1/3 | 1 | 1 | 0,42 | 0,06 |
| A5 | 1/5 | 1/5 | 1/3 | 1 | 1 | 0,42 | 0,06 |

Базою кількісної оцінки програм є не лише аналітично-ієрархічна процедура Сааті, що повсюдно експлуатується для точного визначення вагових коефіцієнтів критеріїв якості. Крім неї, також знайшов застосування метод експертних оцінок, завданням якого є отримання кількісних значень критеріїв якості.

Для оцінного порівняння вибраних криптоалгоритмів проведемо їх порівняльний аналіз за допомогою методу Сааті [1]. Трохи нижче відображено обрані критерії, на підставі яких проводитиметься процедура оцінки:

А1 - Криптостійкість (MIPS);

А2 - Розмір генерованого ключа (до 4096 біт);

А3 - Призначення (шифрування та цифровий підпис);

А4 - Швидкість шифрування (при довжині модуля 1024 біта);

А5 - Швидкість дешифрування (при довжині модуля 1024 біта).

Використовуючи аналітично-ієрархічну процедуру Сааті, встановимо для кожного критерію якості його вагу [2]. Таблиця 3.2 відображає матрицю парних порівнянь, середні геометричні та ваги критеріїв. Правила заповнення матриці парних порівнянь представлені таблиці 3.1.

Таблиця 3.2 Значення коефіцієнтів матриці парних порівнянь

|  |  |
| --- | --- |
| Xij | Значення |
| 1 | перший критерій практично рівноцінний j-му |
| 3 | i-ий критерій меншою мірою важливіше j-го |
| 5 | i-ий критерій важливіший за j-го |
| 7 | Перший критерій переважно важливіше j-го |
| 9 | i-ий критерій набагато важливіший за j-го |

Матриця парних порівнянь, ваги критеріїв та середні геометричні занесені до таблиці 2.

На рисунку 3.1 зображено створену на підставі даних таблиці 2 діаграму вагових коефіцієнтів для критеріїв A 1 , A 2 , A 3 , A 4 та A 5 .

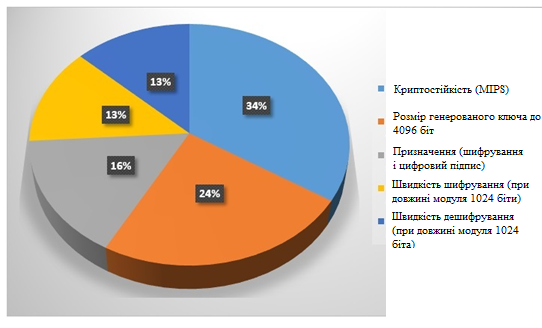


Рисунок 3.1 - Вагові коефіцієнти критеріїв якості

Щоб перевірити матрицю парних порівнянь на несуперечність, зробимо її перевірку. Суми стовпців матриці парних порівнянь [3]:

*R*1 = 1.88; *R*2 = 4.6; *R*3 = 13.67; *R*4=15; *R*5=15.

Після обчислимо додаткову величину *L*, підсумувавши вагові коефіцієнти та добутки сум стовпців матриць: *L* = 5,45.

Таким чином, індекс узгодженості *ІС* = (*LN*)/(*N*-1) = 0,113.

Отже, величина випадкової узгодженості розмірності матриці парних порівнянь: *СлС* = 1,24.

Відношення узгодженості ОС=ІВ/СлС = 0,09 не перевищує 0,2, а отже, додаткове уточнення матриці парних порівнянь не потрібне [4].

Використовуючи обчислені коефіцієнти знайдемо інтегральний показник якості для наступних асиметричних алгоритмів шифрування даних: RSA, DSA, шифросистема Ель-Гамаля, обмін ключами Діффі-Хелмана, протокол Аншеля-Гольдфельда.

Встановимо категоріальну шкалу від нуля до семи (де 0 - якість не задовільно, а 7 - гранично досяжний рівень якості) для встановлення функціональних можливостей вибраних криптоалгоритмів.

Значення вагових коефіцієнтів ai, що відповідають функціональним можливостям аналогів [5]:

1. Криптостійкість (MIPS): a1 = 0.34;
2. Розмір генерованого ключа (до 4096 біт): a2 = 0.24;
3. Призначення (шифрування та цифровий підпис): a3 = 0.16;
4. Швидкість шифрування (при довжині модуля 1024 біта): a4 = 0.13;
5. Швидкість дешифрування (при довжині модуля 1024 біта): a5 = 0.13;

де  [6]. За обраною шкалою визначимо кількісні значення функціональних можливостей  (таблиця 3.3) та обчислимо інтегральні показники якості для обраних асиметричних алгоритмів шифрування:

Таблиця 3.3 Інтегральні показники якості

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Критерії** | **Вагові коефіцієнти** | **Асиметричні алгоритми** | | | | | **Базові значення** |
| RSA | DSA | шифросистема Ель-Гамалю | обмін ключами Діффі-Хелмана | протокол Аншеля-Гольдфельда |
| Криптостійкість (MIPS) | 0,49 | 7 | 7 | 7 | 5 | 5 | 6,2 |
| Розмір генерованого ключа (до 4096 біт) | 0,29 | 5 | 3 | 7 | 3 | 5 | 4,6 |
| Призначення (шифрування та цифровий підпис) | 0,11 | 7 | 7 | 7 | 3 | 3 | 5,45 |

продовження таблиці 3.3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Швидкість шифрування (при довжині модуля 1024 біта) | 0,06 | 7 | 7 | 3 | 5 | 3 | 4,95 |
| Швидкість дешифрування (при довжині модуля 1024 біта) | 0,06 | 3 | 3 | 7 | 3 | 3 | 3,8 |
| Інтегральний показник якості Q | | 6,25 | 5,67 | 6,83 | 4,13 | 4,59 | 5,49 |

де - Інтегральний показник якості для j- го криптоалгоритму.

Побудуємо пелюсткову діаграму інтегрального показника якості кожного криптоалгоритму (рисунок 3.2).

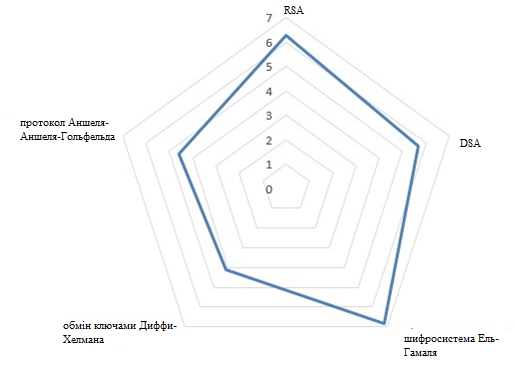


Рисунок 3.2 - Пелюсточна діаграма інтегральних показників якості криптоалгоритмів

Значення характеристик функціональних можливостей (критеріїв) представлені у вигляді пелюсткової діаграми на рисунку 3.3.

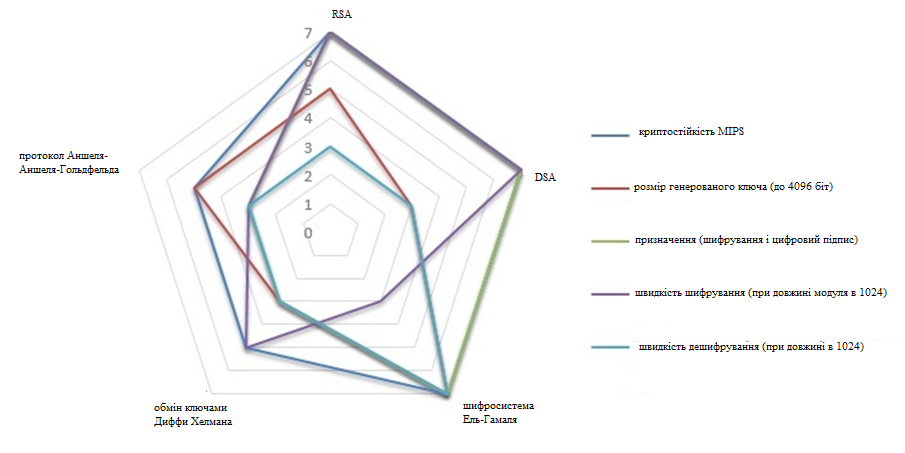


Рисунок 3.3 - Пелюстка діаграма значень функціональних характеристик

Порівняльний аналіз криптоалгоритмів з відкритим ключем показав, що з усіх представлених аналогів жоден не має максимально високих показників за всіма заявленими параметрами, особливо розглянуті криптоалгоритми страждають від низької швидкості шифрації і дешифрації переданих даних, яка обумовлена їх класичною асиметричною структурою [7]. Дана методика експертної оцінки асиметричних криптоалгоритмів дозволила оцінити їх якість з позиції рівня функцій, що реалізуються [8].

3.2 Субекспоненційні алгоритми факторизації

Зробимо спочатку кілька попередніх зауважень. Будемо розглядати лише непарні складові числа *N*. Той факт, що число *N* є складеним, можна встановити за допомогою алгоритмів. Усі описані нижче алгоритми певною мірою узагальнюють ідею алгоритму Ферма. Ще Ж. Лагранж запропонував замість рівностей  використовувати порівняння:

 (3)

І тут власними дільниками числа *N* будуть числа (*х* - *у*, *N*), (*х* + *у*, *N*), якщо *х*: Ф ±*у* (mod *N*).

Ця ідея виявилася плідною та лягла в основу багатьох сучасних методів факторизації цілих чисел. У 1926 р. М. Б. Крайчик описав загальну схему імовірнісного алгоритму факторизації, що складається з чотирьох етапів:

1. Створення деякої множини порівнянь виду:

 (4)

з відносно невеликими *v*.

2. Факторизація чисел *v*, *о*.

3. Почленное перемноження отриманих порівнянь виду (4) з метою отримання порівняння виду (2) з умовою *х*: Ф±у (mod *N*).

4. Знаходження (*х – у*, *N*), (*х + у, N*).

Для обґрунтування такої схеми знаходження дільників числа *N* необхідне таке затвердження.

Нехай *N* - непарне складове число, яке не є ступенем простого числа.

Рівність (2) означає, що *N*|(*х* - *у*)(*х* + *у*). Якщо при цьому (*х* - *у*, *N*) = 1, то *N* | *x* + *у*, тобто *х* = -*у* (mod *N).*

Отже, умова 1 < (*х* ± *у*, *N*) < *N* рівносильна умові *х* Ф ±*у* (mod *N*). Крім того, за умовою затвердження порівняння (3). Тому неважко помітити, що шукана ймовірність Р0 дорівнює відношенню:

*.*

Підрахуємо число розв'язків рівняння z2=1 (mod *N*). Нехай канонічне розкладання N має вигляд   , де по умові затверджено s > 2 і всі числа *р*, непарні. Тоді за китайською теоремою про залишки рівняння z2 = 1 (mod *N*) рівносильне системі рівностей:



Через циклічність груп *Z* рівності  мають рівно по два рішення .   Отже, рівність  має рівно 2s різних за mod *N* рішень. Звідси слідує що:

,

тому що *s* > 2.

Отже, у разі рівності виду (2) не дозволять отримати розкладання числа *N* на нетривіальні співмножники.

Отже, розглядатимемо непарні складові числа N, які є ступенем простого числа.

Зробимо ще кілька зауважень про властивості величини:

.

Для стислості будемо позначати:  через *LN*(*a*). З визначення величини *LN* (*a*) випливає, що виконуються такі рівності:

 (5)

3.3 Алгоритм факторизації на основі еліптичних кривих

Сьогодні сертифікати DigiCert SSL пропонують на вибір два різні алгоритми шифрування - RSA і ECC - вони допоможуть створити в організації більш надійне і масштабоване середовище.

Сертифікат ECC входить до всіх сертифікатів DigiCert Premium SSL абсолютно безкоштовно. ECC надає більш надійний захист та високу швидкодію: на сьогоднішній день цей алгоритм захищає краще, ніж інші методи шифрування, працюючи при цьому з більш короткими ключами (наприклад, 256-розрядні ключі ECC дають той самий захист, що й 3072-розрядний ключ RSA). Більш надійний захист, здатний впоратися зі стрімким зростанням числа підключень з мобільних пристроїв та планшетів. У міру підвищення рівнів захисту розмір ключа ECC збільшується повільніше, ніж у разі використання інших методів шифрування, що дозволяє продовжити термін експлуатації наявного обладнання та збільшити віддачу від інвестицій[2-4]. Кореневі сертифікати DigiCert ECC не застаріють ще як мінімум п'ять років: ви можете бути впевнені, що ваш сертифікат ECC діятиме для всієї екосистеми вашої організації Схвалено урядом США: ECC сертифікований FIPS (Федеральний стандарт обробки інформації в США) та підтримується Агентством національної безпеки США[5-9].

Більш високий рівень безпеки за рахунок того, що сертифікати з алгоритмом ECC з 256-бітним ключем в 10000 разів більш стійкі до злому, ніж сертифікати з алгоритмом RSA з 2048-бітним ключем, і безпеки еквівалентні сертифікатам RSA з 3072-біт.

Підвищення продуктивності серверів, особливо в періоди пікового навантаження, за рахунок можливості обробляти більшу кількість запитів при меншому завантаженні ЦПУ, що стає все більш важливим у міру поширення планшетів та мобільного інтернету та, отже, підвищення вимог до продуктивності веб-інфраструктури.

Підвищення швидкості обробки запитів та зниження часу відгуку. Внутрішні тести DigiCert показали, що сервер з RSA-сертифікатом може обробляти 450 запитів в секунду із середнім часом відгуку в 150 мілісекунд, тоді як у сервера з ECC-сертифікатом при роботі в тих же умовах середній час відгуку - приблизно 75 мілісекунд [10].

ECC забезпечує найкращу масштабованість та дозволяє здійснювати одночасну взаємодію мільярдів хостів, суттєво підвищуючи ефективність обміну інформацією у віртуальних середовищах, використання хмарних рішень та глобальної електронної торгівлі. З точки зору користувача, покращена обчислювальна продуктивність та ефективне використання інфраструктури створюють більш сприятливі умови для роботи та збільшують продуктивність[11-14].

Факторизація - розкладання складеного числа N на множники. Формально алгоритм, запропонований Ленстрою, містить наступні кроки.

Крок 1. Вибирається ЕК з поля порядку N, тобто. цілі числа з діапазону 1<*b,x*1,*y*1<*N.*

Крок 2. Розглядаємо кубічну криву: *E*:*y*2=*x*3bx+c;=y12-x13-*bx*(mod *N*);*b,cZ*, базисну точку ЕК =(x1,y1) та базисну точку ЕК.

Крок 3. Обчислюємо (4*b*3+27*c*2,*N*); цим перевіряється, чи є редукції кривої Е еліптичними. Якщо НСД = *N*, перехід до Кроку 1 та вибір нового *b*. Якщо 1<(4*b*3+27*c*2,*N*)=1 то, перехід до кроку 4.

Крок 4. Вибираємо число *k*, що є добутком невеликих простих чисел у невеликих ступенях: *k* = НСК (2,3, …, *М*), де *М*-натуральне число.

Крок 5. Обчислюємо кратну базисну точку *kP*=(*ak*/*e*2*k*;*bk*/*e*3*k*).

Крок 6. Обчислюємо *D*=(*ek,N*). Якщо 1<*D*<*N*, то *D*-нетривіальний дільник *N*; якщо *D* =1, то перехід до кроку 4 і збільшуємо *k*, або до кроку 1 і вибір нової ЕК.

На кроці 5 обчислюється точка *kP*. Її можна вирахувати найбільше за: 2log2*k* кроків, тобто складання точок на кривій. Більше того, при обчисленнях використовуються лише значення *k* (mod *N*).

Наведемо реалізацію алгоритму на мові Python:

import math

from fractions import gcd

import random

def L\_elliptic\_curve\_factor(count):

""" Elliptic curve factoring method """

if count <=0:

raise Exception("Integer %s must be possitive " % count)

# Can't be 1 and can't factor a prime!

if 1 <= count <= 2 or is\_prob\_prime(count):

return [count]

# random point in the plain (values less than count)

p0, t0 = random.randrange(1, count), random.randrange(1, count)

factors = list()

bound = int(math.sqrt(count))

for a in xrange(2, count):

# Build curve out of random points

b = t0\*\*2 - p0\*\*3 - a\*p0

# Check curve is not singular

if 4\*a\*\*3 - 27\*b\*\*2 ==0:

continue

# Initially double point

s = 3\*p0\*\*2 + a

(p,t) = (s\*\*2 - 2\*p0, s\*((s\*\*2 - 2\*p0) - p0) - t0)

# Keep adding points until gcd(p-p0, count) != 1

for k in xrange(2,bound):

for i in xrange(0,math.factorial(k)):

d = gcd(p- p0, count)

if d != 1:

return L\_elliptic\_curve\_factor(int(d)) + L\_elliptic\_curve\_factor(int(count /d))

else:

# Point doubling arithmetic

s = (t - t0) \* modInv(p - p0, count)

p = s\*\*2 - p - p0

t = - t + s \* (s\*\*2 - p - p0 - p)

Звичайний підхід - багаторазове підсумовування точки із собою. Ленстра зауважив, що у кільці існує неможливість обчислити суму точок (отримати нейтральний елемент O), де необхідно обчислювати коефіцієнт =(2-1)/(2-1), виникає розподіл на нуль частіше ніж у полі, що може призводити до нейтрального елемента швидше [7-9]. Справді, різниця: 2-1(mod *N*) звертається в нуль при значеннях кратних як самому N, але за кратних його дільників.

Приклад Нехай задано складене число: *N*=246082373. Вибиремо **

Представимо ** розкладанням за ступенями 2. Обчислюватимемо:

**

**

**

**

**

Відповідно:

.

Тепер, обчислюємо:

.

Отримуємо одиницю (рішення немає) і переходимо до кроку 1 для вибору нового значення *k*, яке збільшимо в 7 разів.



Представимо нове значення k степенем 2.



Обчислимо:  і редукуємо за модулем N, .

Скористаємось алгоритмом Евкліда.  і получаємо  нетривіальний множник 

Факторизацією цілого числа називається його розкладання в добуток простих співмножників. Таке розкладання, згідно з основною теоремою арифметики, завжди існує і є єдиним (з точністю порядку множників).

Будемо позначати число, яке потрібно факторизувати буквою *N*, а два співмножники, в результаті множення яких отримаємо *N*, як *a* і *b*.

Алгоритм факторизації Ленстри (факторизація на основі використання еліптичних кривих) є одним з найшвидших методів факторизації. Він має багато спільного з методом Полларда (*p* - 1), але працює значно швидше, слід відзначити, що він є субекспоненційним методом. Головна особливість алгоритму - це те, що час, витрачений на факторизацию, залежить не від розмірності *N*, а від розміру найменшого дільника числа *N*[1].

Виберемо деяке значення *B*, яке є максимальною межею числа-подільника *N*.

Згенеруємо випадковим чином значення *x, y, a,* які належать множині цілих чисел від 0 до *N*-1. Ці значення визначають криву, а так само *x* і *y* визначають початкову точку *P*.

Обчислимо *b* = *y*2-*x* 3-*ax* mod *n* і *g* = Н.С.Д. (*N*, 4*a*3 + 27*b*2). Важливо, що б *g* не дорівнює *N*, інакше повертаємося до початку. Якщо 1 <*g* <*n*, тоді припинимо обчислення - дільник знайдений. Цей варіант можливий при малих значеннях числа *N* (наприклад 10), при зростанні *N* ця можливість зустрічається все рідше.

Далі потрібно виконати цикл, в результаті якого для кожного простого числа *p* (в межах від 2 до *B*-1) обчислимо точку *P*, домножену на *p* *r*, де *r* найбільша ступінь, яка задовольнить умові *p* *r* <*B*.

У арифметиці полів немає операції ділення, яка необхідна в формулах для знаходження координат точок, так що перетворимо вираз у вигляді множення, а зворотний елемент шукаємо по розширеному алгоритму Евкліда. До того ж кожне нове обчислення обчислюємо по модулю числа *N*.

Алгоритм завершить свою роботу, коли буде знайдений Н.С.Д. (*N, P*), який більше 1 і менший *N*. Щоб уникнути помилки у відповіді, функція знаходження Н. С.Д. повинна перевіряти тільки позитивні значення[2].

Розглянемо графік часу, який отримано при виконанні факторизації різних чисел.

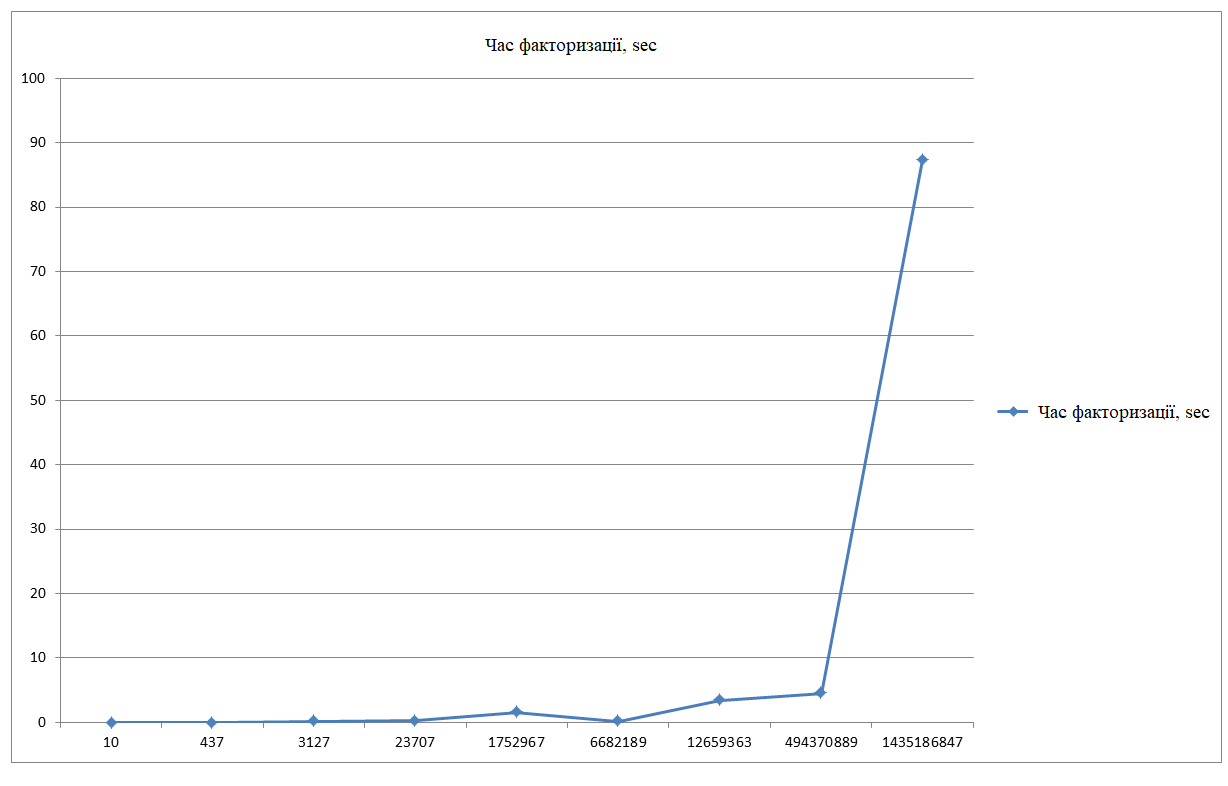


Рисунок 3.4 - Результат наведено з експериментальної програми

*N* = 10 - 0,005 sec;

*N* = 437 - 0,019 sec;

*N* = 3127 - 0,055 sec;

*N* = 23707 - 0,191 sec;

*N* = 1752967 - 1,534 sec;

*N* = 6682189 - 0,143 sec;

*N* = 12659363 - 3,376 sec;

*N* = 494370889 - 4,484 sec;

*N* = 1435186847 - 87,377 sec;

Так як крива генерується з випадкових значень, то час факторизації при кожному новому запуску програми буде різним. Для достовірності експерименту, необхідно взяти ці ж числа і виконати факторизацию ще кілька разів і звіримо результати.

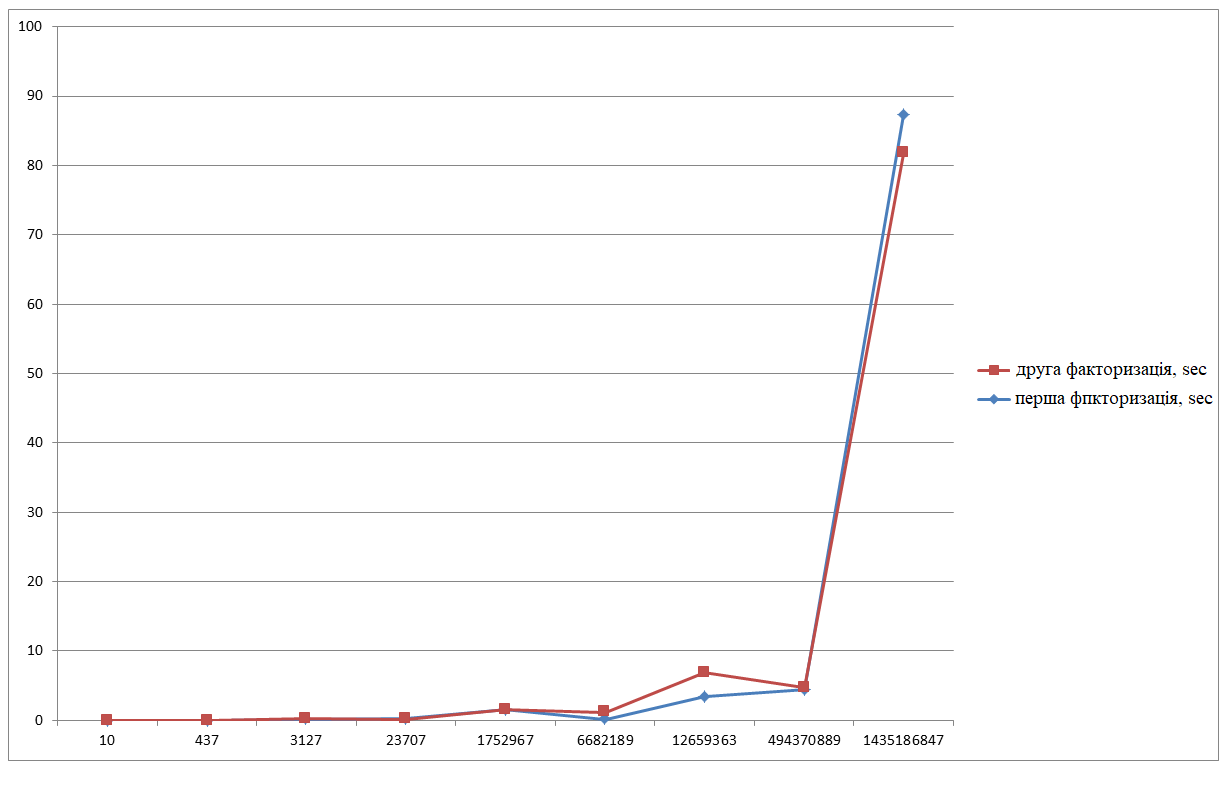


Рисунок 3.5 - Результати повторної факторизації

N = 10 - 0,016 sec;

N = 437 - 0,016 sec;

N = 3127 - 0,218 sec;

N = 23707 - 0,079 sec;

N = 1752967 - 1,484 sec;

N = 6682189 - 1,125 sec;

N = 12659363 - 6,906 sec;

N = 494370889 - 4,781 sec;

N = 1435186847 - 81,766 sec;

Проведено також третій експеримент по факторизації з використанням вдосконаленого алгоритму Ленстри(рисунок 3.6).

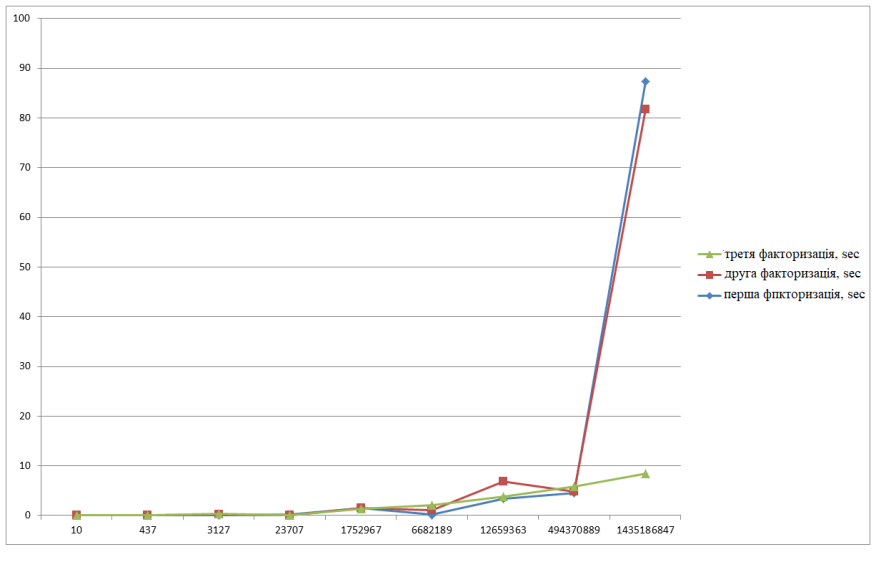


Рисунок 3.6 Результати третього експерименту факторизації

N = 10 - 0,012sec;

N = 437 - 0,022 sec;

N = 3127 - 0,156 sec;

N = 23707 - 0,205 sec;

N = 1752967 - 1,418 sec;

N = 6682189 - 1,056 sec;

N = 12659363 - 0,25 sec;

N = 494370889 - 5,488 sec;

N = 1435186847 - 14,117 sec;

Як ми бачимо на останньому графіку, час витрачений на останнє число помітно зменшився, це характеризується тим, що крива генерується випадковим чином. Слід зауважити, що час, витрачений на виконання алгоритму, залежить і від значення *B*, яке вводимо. В експерименті було обрано значення наближені до значень двох співмножників (*a* і *b*), але якщо виконувати операцію над числом, де початкові множники невідомі навіть приблизно, то варто вибирати межу вище. Але чим вище межа, тим більше часу піде на обчислення точок, тому що їх буде більше, це потрібно враховувати.

ВИСНОВКИ

Досліджено теорія еліптичного шифрування, проведено оцінку асиметричного шифрування на еліптичних кривих. У роботі наводиться дослідження основних кроків та етапів алгоритмів квадратичного решета, ферма подібних алгоритмів для факторизації чисел.

Проведено дослідження існуючих підходів до факторизації, що дало можливість виділити перспективні напрямки для дослідження, а саме «природні алгоритми» та алгоритми фпкторизації на основі еліптичних кривих.

Проведено дослідження та розробку системи для криптоаналізу використовуючи «природні алгоритми». Експериментальним чином було оцінено їхню ефективність.

Досліджено математичні основи алгоритмів для ефективної генерації еліптичних кривих. Досліджено можливості криптографії та крипто аналізу на основі еліптичних кривих.

Розроблено програмний засіб для факторизації на основі еліптичних кривих, що підтверджує ефективність запропонованих алгоритмічних рішень. Алгоритм Ленстри для факторизації багаторозрядних чисел на основі еліптичних кривих має досить багато кроків, що надають можливості оптимізації обчислювальних процесів. Проведена програмна реалізація алгоритму Ленстри для факторизації багаторозрядних чисел на основі еліптичних кривих.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Lenstra Н. W. Factoring Integers with Elliptic Curves // Ann. Math. —1987. - № 126. - p.649-673.
2. Voorhoeve M. Factorization algorithms of exponential order // Computational methods in number theory. V. 1 / H.W. Lenstra and R. Tijdeman, editors. Amsterdam, 1982. P. 79—88.
3. Ишмухаметов Ш.Т. Методы факторизации натуральных чисел: учебное пособие / Ш.Т. Ишмухаметов.– Казань: Казан. ун. 2011.– 190 с. ,c. 52
4. Николайчук Я.М. Фундаментальні засади теорії факторизації багаторозрядних чисел на основі фракталів зображень в околі рішення//, Івасьєв С.В./ Збірник матеріалів міжнародної наукової координаційної наради «Інформаційні проблеми комп’ютерних систем, юриспруденції, енергетики, економіки, моделювання та управління»(ICSM) – Тернопіль, 2014р – С. 116-120.
5. Болотов А.А. и др. Элементарное введение в эллиптическую кривую: Алгебраические и арифметические основы. – Москва: Ком. Книга, 2006г. – 328с.
6. Ван дер Варден Б.Л. Алгебра — Москва: Наука, 1976г
7. Клеменс Г. Мозаика теории комплексных кривых. — Москва: Мир, 1984г. – 160с.
8. Кнэпп Э. Эллиптические кривые. – М.: Изд. «Факториал Пресс», 2004. – 488с.
9. Мамфорд Д. Алгебраическая геометрия. Комплексные проективные многообразия. — Москва: Мир, 1979г
10. Koblitz N. Elliptic Curve Cryptosystems// Mathematics of Computation. — 48. 1987. — p. 203 — 209.
11. Рид М. Алгебраическая геометрия для всех. — Москва: Мир, 1991г.
12. Соловьёв Ю.П. и др. Эллиптические кривые и современные алгоритмы теории чисел. Москва – Ижевск: Институт компьютерных исследований, 2003г. – 192с.
13. Степанов С.А. Арифметика алгебраических кривых. – Москва: Наука, 1991г. – 368с.
14. MenezesA.Elliptic Curve PublicKeyCryptosystems. Boston:KluwerAcademic Publishers,1993.- p. 126.
15. Silverman J. The Arithmetic of Elliptic Curves. — New York: Springer, 1986. p. 400.
16. Zemor. Cours de cryptographie Vuibert, 2000, — p. 212. УКНД 35.040
17. A.K. Lenstra, H.W. Lenstra, Jr. The Factorization of the Ninth Fermat Number / A.K. Lenstra.- New York City: ACM, 1993. — p.246.
18. Функция односторонняя [Электронный ресурс] / Математическая криптография - Режим доступа : http://cryptography.ru/ref/функция\_одностороння/- 09.01.2017 р.
19. Simmons G. I. "Cryptology", Encyclopedia Britannica, 16th edition / G. I. Simmons - Sandia National Laboratories 1986. - p.913.
20. Анохін М. А., Варновскій Н. П. Криптографія в банківській справі / М. А. Анохін.-M:МІФІ, 1997.- 174 c.
21. Ишмухаметов Ш.Т., Методы факторизации натуральных чисел [Текст] : учеб. пособие / Ш.Т. Ишмухаметов.– Казань:Казан.ун., 2011.- 201с.
22. Герман О.Н. Теоретико-числовые методы в криптографии[Текст] / О.Н. Герман, А.Ю. Нестеренко. — М.:Aкадемия, 2012. — 300 с.
23. Bressoud D. M. Factorization and Primality Testing [Text] /D. M. Bressoud. — New York: Springer-Verlag, 1989. — 260 p.
24. Richard P.Brent Some parallel algorithms for integer factorisation [Text] / P. Richard. — Oxford:Oxford University, 1999. — 27p.
25. Lenstra A. K. Factoring polynomials with rational coefficients [Text] / A. Lenstra, H. Lenstra, S. Lovaz.-Luxembourg: Springer Science Business Media, 1982. — 208 p.
26. Василенко О. Н. Теоретико-числовые алгоритмы в криптографии / О. Н. Василенко. – М.: МЦНМО, 2003. – 328 с.
27. М.Г. Адигеев Введение в теорию сложности [Текст] / М.Г. Адигеев.- Ростов-на-Дону:РГУ, 2004.-35с.
28. Эббинхауз Г.Д. Машины Тьюринга и рекурсивные функции/ Г.Д. Эббинхауз. – М.: Мир, 1972. – 264 с.
29. Томас Х. Алгоритмы. Построение и анализ [Текст]/ Х. Томас - М.:Вильямс, 2016.-1328с.
30. Роман Душкин Квантовые вычисления и функциональное программирование / Р. В. Душкин.-М.: ДМК Пресс, 2015. – 232 с.
31. В.Н. Ручкин Естественный параллелизм квантовых компьютеров и нейровычислителей/ В.Н. Ручкин, В.А. Романчук, В.А. Фулин.- Рязань.:Рязанский государственный университет им. С.А. Есенина, 2013 – 387 с.
32. С.А. Дуплий. Квантовая информация, кубиты и квантовые алгоритмы / С. Дуплий, В. Калашников, Е. Маслов. – Харьків.:Харьківський національний університет ім. В. Н. Каразіна, 2005.- 217 с.
33. Дж.Прескилл Квантовая информация и квантовые вычиселения / Д. Прескилл.-М.:Ижевск, 2008. - 464 с.
34. П.А. Правильщиков Квантовый параллелизм и решение уравнений в задачах управления на базе новой модели вычислений / П.А. Правильщиков.- М.:Институт проблем управления им. В.А. Трапезникова, 2014.- 179 с.
35. Wootters W.A Single Quantum Cannot Be Cloned [Text] / Wootters W., Zurek W.//Nature.-1982.-№5886.-803p.
36. Bennett C. Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels [Text] /Bennett C., Brassard G., Crépeau C.//Physical review letters.-1993.-№13.-1895-1899pp.
37. А.Г. Грозин Квантовый компьютер для чайников/ А.Г. Грозин.-  
    Новосибирск.: Препринт, 2004.-240 с