Одеський національний університет імені І.І.Мечникова
(повне найменування вищого навчального закладу)

Факультет математики, фізики та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерної алгебри та дискретної математики
(повна назва кафедри (предметної, циклової комісії))

Дипломна робота

на здобуття ступеня вищої освіти «бакалавр»
(освітньо-кваліфікаційний рівень)

на тему: Цифровий підпис на еліптичній кривій
(назва українською)

Digital signature on an elliptic curve
(назва англійською)

Виконав: студент денної форми навчання
спеціальності 123 Комп'ютерна інженерія.
(шифр і назва напряму підготовки, спеціальності)
Щупак Тимур Олександрович
(прізвище, ім'я, по-батькові)

Керівник доктор фізико-математичних наук,
професор Варбанець П.Д.
(науковий ступінь, вчене звання, прізвище та ініціали, **підпис**)

Рецензент_____
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
№___ від «___»_____ 2023 р.

Завідувач кафедри
_____ _О.В. Савастру_
(підпис) (прізвище, ініціали)

Захищено на засіданні ЕК № ___
протокол № __ від «___» ____ 2023 р.
Оцінка _____/_____/_____
(за національною шкалою, шкалою ECTS, бали)
Голова ЕК
_____ _Н.Ф. Казакова_
(підпис) (прізвище, ініціали)

Одеса – 2023

ABSTRACT

The diploma thesis explores the topic of "Digital Signature on Elliptic Curves." The paper discusses the principles and mathematical aspects associated with elliptic curves: curve points, addition and multiplication operations, properties of discrete logarithm on elliptic curves.

An overview of existing algorithms for digital signature on elliptic curves, specifically ECDSA (Elliptic Curve Digital Signature Algorithm), its advantages, disadvantages, and potential application areas, has also been conducted.

Based on the information obtained through the research, in the practical part of the thesis, an algorithm for digital signature on elliptic curves was developed, taking into account specific requirements and utilizing modern mathematical and cryptographic methods.

The main result of the thesis is the development of an algorithm for digital signature on elliptic curves, and its efficiency was also considered.

АНОТАЦІЯ

У дипломній роботі розглядається тема «Цифровий підпис на еліптичній кривій». У роботі розглянуті принципи та математичні аспекти, які пов’язані з еліптичними кривими: точки кривої, операції додавання та множення, властивості дискретного логарифму на еліптичних кривих.

Також був проведений огляд існуючих алгоритмів цифрового підпису на еліптичних кривих, а саме ECDSA (Elliptic Curve Digital Signature Algorithm), його переваги, недоліки та потенційні сфери застосування.

За допомогою інформації отриманої в результаті виконання роботи в практичній частині роботи був розроблений алгоритм цифрового підпису на еліптичних кривих з урахуванням конкретних вимог та використовуючи сучасні математичні та криптографічні методи.

Основним результатом роботи був розроблений алгоритм цифрового підпису на еліптичних кривих, також була розглянута його ефективність.

ЗМІСТ

ВСТУП

У сучасному інформаційному суспільстві з поступовим зростанням інформаційних технологій, обчислювальних технологій та технологій Інтернету, які широко використовуються в усіх сферах суспільства і поступово проникають в усі сфери життя, зростає попит на безпечну, ефективну і повну передачу інформації.

Однак різноманітні інформаційні системи, побудовані на цій основі, приносять багато зручностей і необмежених бізнес-можливостей в життя та роботу людей, але вони також повні ризиків і небезпек. Оскільки Інтернет вразливий до атак, це може призвести до витоку інформації і спричинити значні збитки.

Інформаційні технології стали важливою складовою загальної національної потужності, тому інформаційна безпека стала гарантією забезпечення здорового і регламентованого розвитку національного економічного інформаційного будівництва. Технологія інформаційної безпеки увійшла в новий період швидкого розвитку в умовах стрімкого розвитку інформаційних технологій, сформувавши низку категорій технологій захисту, таких як криптографічні технології, технології довіреної обчислювальної техніки, технології захисту від електромагнітного випромінювання, технології виявлення вторгнень у систему та технології виявлення та усунення комп'ютерних вірусів.

Існує багато технологій мережевої безпеки, і на сьогоднішній день ключовою технологією, необхідною у електронній комерції, електронному урядуванні, електронному банківському обслуговуванні та електронній пошті, є цифрові підписи.

Цифровий підпис може забезпечити автентифікацію особи, цілісність даних, недоступність відмови від підпису та інші функції, що є ключовими

технологіями забезпечення цілісності та автентифікації інформації, а також ключовими технологіями електронної комерції та мережевої безпеки.

Одним з найефективніших методів безпеки є цифрові підписи з використанням еліптичних кривих. Еліптичні криві – математичні об’єкти, які використовуються в криптографії для розробки алгоритмів шифрування, цифрових підписів та інших криптографічних протоколів.

Метою цієї дипломної роботи є детальне вивчення та дослідження алгоритмів цифрового підпису на еліптичних кривих, зокрема алгоритми ECDSA(Elliptic Curve Digital Signature Algorithm), та ElGamal. ECDSA є одним з найпоширеніших алгоритмів цифрових підписів. Він базується на математичних властивостях еліптичних кривих.

У процесі роботи будуть досліджені основні математичні принципи еліптичних кривих, а також розглянуті принципи функціонування алгоритму, та його застосування для цифрових підписів.

Основними цілями роботи є:

- розуміння математичних основ еліптичних кривих та їхніх криптографічним властивостей;
- дослідження алгоритму ECDSA та ElGamal. Їх оцінка, переваги та обмеження;
- розробка власного алгоритму цифрового підпису на еліптичних кривих з урахуванням сучасних вимог безпеки та ефективності.

Результати роботи можуть мати практичне застосування у сферах, які потребують надійного забезпечення цифрових підписів, такі як: електронна комерція, фінансові транзакції та безпека інформаційних систем.

1 ОСНОВНІ ПОНЯТТЯ ЦИФРОВОГО ПІДПИСУ

Визначення цифрового підпису:

Цифровий підпис - це електронна форма підпису, що підтверджує автентичність та цілісність електронного документа або повідомлення. Використовуючи криптографічні методи, цифровий підпис гарантує, що дані не були змінені після підпису.

Криптографічні компоненти цифрового підпису:

- Хеш-функції: Вони генерують унікальний хеш або криптографічне значення для підписуваного документа. Хеш-функція служить як контрольна сума, яка дозволяє виявити навіть найменші зміни в документі;

- Ключі: Цифровий підпис використовує два ключі - приватний і публічний. Приватний ключ використовується для створення підпису, тоді як публічний ключ використовується для перевірки підпису (рис. 1.1);

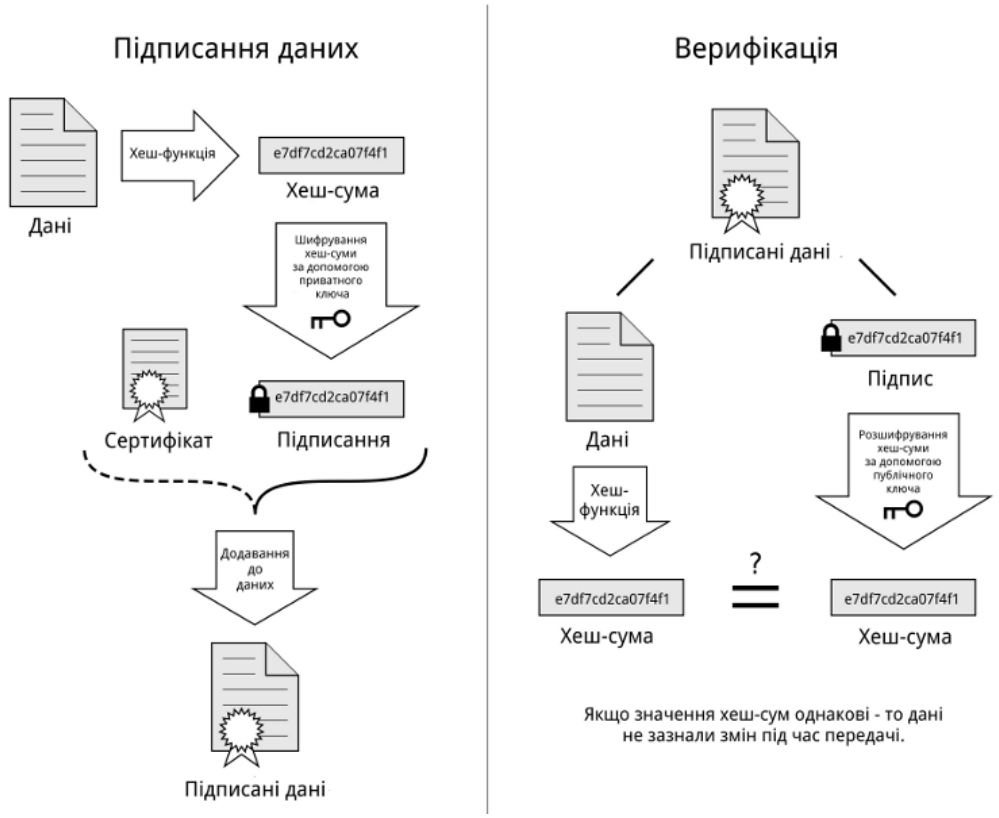


Рисунок 1.1 – Схема зображення роботи цифрового підпису

- Криптографічні алгоритми: Цифровий підпис базується на різних криптографічних алгоритмах, таких як RSA, DSA або ECDSA, які забезпечують безпеку та невідмовність підпису;

Принципи цифрового підпису:

– Протидія підробці: ніхто не може підробити підпис на повідомленні, оскільки приватний ключ відомий тільки особі, яка підписує повідомлення. Очевидно, що власник приватного ключа повинен добре зберігати свій власний ключ, так само, як він зберігає ключ від своєї двері.

– Неможливість втручання: у випадку цифрового підпису підпис і оригінальний документ утворюють нерозривну цілість, яку не можна втрутитися, тим самим забезпечуючи цілісність даних.

– Захист від повтору: цифрові підписи можуть бути захищені від повторних атак шляхом додавання порядкового номера, мітки часу тощо до підписаного повідомлення.

– Протидія відмові визнати підпис: цифрові підписи можуть бути ідентифіковані і не можуть бути підроблені, тому допоки підписане повідомлення зберігається, це подібно до збереження вручну підписаного договору, тобто підписник не може відмовитися визнати його, якщо він зберігає докази. Що, якщо одержувач отримує підписане повідомлення від іншої сторони, але заперечує його отримання? Для запобігання відмові визнати повідомлення одержувачем, система цифрового підпису передбачає повернення одержувачем підписаного повідомлення іншій стороні, третій стороні або використання механізму залучення третьої сторони. Таким чином, жодна зі сторін не може заперечити повідомлення, що стосується теорії невизнання підписів.

– Конфіденційність: Забезпечуючи конфіденційність, атаки перехоплення стають неефективними. Вручну підписані документи, такі як контрактні тексти, не мають конфіденційності, і якщо документ загубиться, існує великий ризик витоку інформації. Цифрові підписи, які шифрують

повідомлення для підпису, пов'язані з теорією шифрування або криптографією.

Цифрові підписи можуть бути використані в техніках аутентифікації і можуть бути використані для наступних типів аутентифікації:

– Аутентифікація сутності: Перед тим, як повідомлення буде передано, використовується ідентифікований протокол для аутентифікації того, що зв'язок відбувається між погодженими комунікаційними сутностями.

– Аутентифікація повідомлення: Після аутентифікації сутності, обидва комунікуючі суб'єкти можуть спілкуватися один з одним. Щоб забезпечити достовірність даних, повідомлення повинно бути аутентифіковане, тобто отримувач повинен мати змогу перевірити автентичність походження, часу та призначення повідомлення. Це також, як правило, досягається за допомогою технології цифрового підпису.

– Аутентифікація користувача: Аутентифікація користувача є першою лінією захисту багатьох застосунків, щоб запобігти доступу до даних незаконними користувачами. Крім контролю паролю, використання технології цифрового підпису в процесі аутентифікації безперечно підвищує рівень контролю доступу.

Переваги цифрового підпису:

- Після підписання документа відправник не може заперечувати свою участь у процесі підпису;

- Цифровий підпис дозволяє ефективно підписувати та перевіряти багато документів без необхідності фізичної зустрічі;

- Цифровий підпис дозволяє перевірити, що дані були підписані відповідним відправником;

- Цифровий підпис забезпечує виявлення навіть найменших змін або підробок в підписаному документі.

Основні поняття цифрового підпису, такі як його визначення, принципи та криптографічні компоненти, відіграють ключову роль у забезпеченні безпеки електронної комунікації та автентифікації даних.

Цифровий підпис є потужним інструментом, що дозволяє підтверджувати автентичність, невідмовність та цілісність електронних документів, сприяючи розвитку цифрового суспільства.

1.1 Опис цифрового підпису на RSA

Алгоритм RSA, який був створений у 1977 році Рональдом Райвестом, Аді Шаміром та Леонардом Адлеманом, є криптографічним методом, що базується на складних операціях з великими простими числами. Його головна ідея полягає в унікальності факторизації великих цілих чисел.

Алгоритм RSA використовує два ключі - приватний і публічний. Приватний ключ є секретним і використовується для створення цифрових підписів, тоді як публічний ключ розповсюджується і використовується для перевірки підпису. Публічний ключ складається з двох чисел - модуля (n) та показника (e), а приватний ключ містить той самий модуль (n) та інший показник (d).

Для створення цифрового підпису на RSA використовується приватний ключ. Спочатку повідомлення, яке треба підписати, конвертується у числове представлення або використовується хеш-функція. Потім застосовується математична операція підпису з використанням приватного ключа, яка дає результат - цифровий підпис.

Для перевірки цифрового підпису використовується публічний ключ. Підписане повідомлення разом з цифровим підписом передаються отримувачу. Отримувач перетворює повідомлення в числове представлення або використовує хеш-функцію, а потім використовує публічний ключ для здійснення математичних операцій перевірки підпису. Якщо підпис співпадає з підписом, отриманим після перевірки, то цифровий підпис вважається валідним.

1.2 Опис цифрового підпису по алгоритму ElGamal

Алгоритм Ель-Гамала, який використовується для створення цифрових підписів, є одним з методів криптографії, що забезпечує автентичність та цілісність електронних документів та повідомлень. Цей алгоритм був розроблений Диффі-Хеллманом та Ель-Гамалем у 1985 році.

Його основна ідея полягає у використанні математичних операцій з простими числами та групами для створення цифрових підписів. Алгоритм базується на складності проблеми дискретного логарифму.

У алгоритмі Ель-Гамала також використовуються два ключі: приватний і публічний. Приватний ключ є секретним і використовується для створення підпису, тоді як публічний ключ поширюється та використовується для перевірки підпису.

Для створення цифрового підпису за алгоритмом Ель-Гамала використовується приватний ключ. Спочатку повідомлення, яке потрібно підписати, перетворюється на числове представлення. Потім застосовуються математичні операції, такі як піднесення до степеня та множення, відповідно до правил алгоритму Ель-Гамала, що призводить до отримання цифрового підпису.

Для перевірки цифрового підпису використовується публічний ключ. Підписане повідомлення разом з цифровим підписом передається отримувачу. Отримувач застосовує математичні операції з використанням публічного ключа згідно з правилами алгоритму Ель-Гамала для перевірки підпису. Якщо підпис збігається з підписом, отриманим після перевірки, то цифровий підпис вважається дійсним.

Алгоритм Ель-Гамала має кілька переваг, таких як можливість використання одного публічного ключа для багатьох підписів та стійкість до атак, пов'язаних з використанням квантових комп'ютерів. Проте, він також має свої обмеження та недоліки, зокрема пов'язані з більшою обчислювальною складністю порівняно з іншими алгоритмами.

2 ЕЛІПТИЧНІ КРИВІ ТА ЇХ ЗАСТОСУВАННЯ У КРИПТОГРАФІЇ

Еліптичні криві використовуються в сучасній криптографії для реалізації різноманітних криптографічних протоколів і алгоритмів. Вони базуються на математичних властивостях еліптичних кривих, які дозволяють здійснювати ефективні розрахунки з криптографічною метою. Приклад еліптичних кривих наведений в рис.2.1.

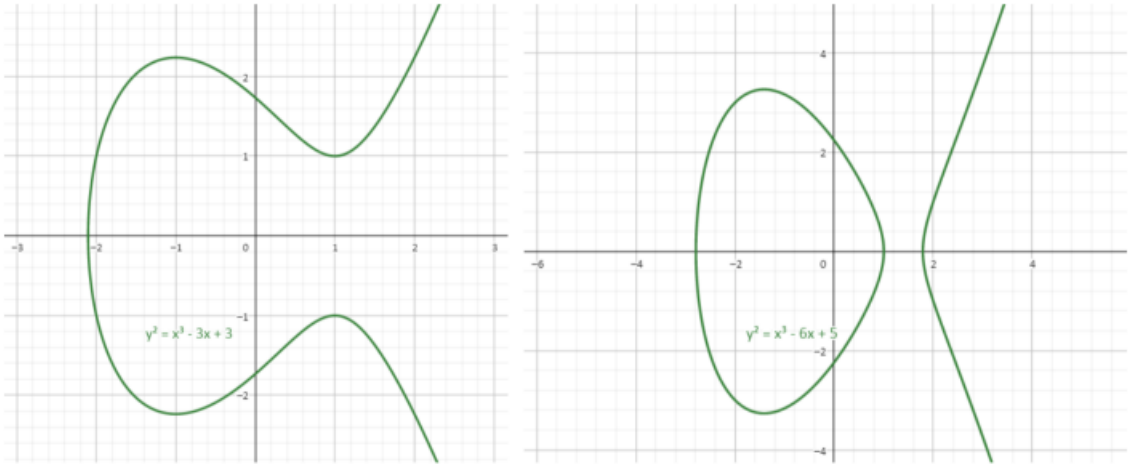


Рисунок 2.1 – Приклад еліптичних кривих.

Система криптографії на еліптичних кривих є відносно новою технологією порівняно з RSA та іншими криптосистемами. У системі криптографії на еліптичних кривих першою проблемою є вибір безпечної еліптичної кривої. Вибір базової точки також є ключовим елементом у побудові системи криптографії на еліптичних кривих.

Безпека систем криптографії на еліптичних кривих над скінченними полями базується на невирішеності проблеми дискретного логарифма в групі точок еліптичних кривих над скінченними полями. На сьогоднішній день найефективніші алгоритми для вирішення цієї математично складної

проблеми все ще вимагають часу, що зростає експоненційно. В той же час, оскільки вимагані довжини ключів у криптографічних схемах стають все складнішими для реалізації, було виявлено, що криптографічні схеми на основі еліптичних кривих є ефективним рішенням для подолання цієї складності, що робить їх гарячою темою дослідження.

Одним з найвідоміших криптографічних протоколів, що використовують еліптичні криві, є Elliptic Curve Cryptography (ECC). В порівнянні з іншими криптографічними методами, такими як RSA, ECC забезпечує високий рівень безпеки при використанні коротших ключів.

Schoof уперше запропонував алгоритм для обчислення кількості раціональних точок на еліптичних кривих у 1989 році, а Атікін та Елкіс розробили алгоритм для обчислення кількості раціональних точок на еліптичних кривих також у 1989 році. Атікін та Елкіс покращили алгоритм між 1989 і 1992 роками, а потім Кувергн, Монрейн, Лерсьє та інші його поліпшили так, що кількість раціональних точок на еліптичних кривих, які задовольняють криптографічним вимогам, можна легко обчислити до 1995 року. Проблема вибору еліптичних кривих та обчислення кількості раціональних точок на довільній еліптичній кривій була вирішена.

Використання еліптичних кривих в криптографії має кілька переваг. По-перше, вони дозволяють ефективну обробку даних з високою криптографічною міцністю. По-друге, вони знижують вимоги до обчислювальних ресурсів, таких як пам'ять і обчислювальна потужність. Це особливо важливо при використанні еліптичних кривих в пристроях з обмеженими ресурсами, наприклад, вбудованих системах.

Опис еліптичних кривих включає параметри кривої, такі як форма і розташування точок на площині. Також визначається базова точка, яка використовується для генерації ключів та виконання криптографічних операцій. Для виконання різних криптографічних операцій, наприклад, шифрування або підписування, використовуються спеціальні алгоритми, такі як додавання, множення та подвоєння точок на кривій.

Важливо правильно вибирати параметри кривої та використовувати відповідні алгоритми для забезпечення безпеки. Існують стандарти, які визначають рекомендовані параметри для використання в еліптичній криптографії.

З-поміж багатьох алгоритмів, еліптична криптосистема широко використовується завдяки своїй короткій довжині ключа, швидкому цифровому підпису, невеликому обсягу обчислювальних даних, швидкості операцій та хорошій гнучкості за однакових умов безпеки. Завдяки високому рівню безпеки, що досягається за допомогою ECC, потребується лише невеликий додатковий обсяг обчислень та затримок. Невеликий додатковий обсяг відображається в обсязі обчислень, зберіганні, пропускній здатності, розмірі апаратного та програмного забезпечення тощо, а затримка відображається у швидкості шифрування або підтвердженні підпису. Тому ECC особливо корисна у випадках, коли є обмеження щодо обчислювальної потужності та простору інтегральної схеми (наприклад, у IC-картах) та пропускної здатності (наприклад, у високошвидкісних комп'ютерних мережах).

Узагальнюючи, еліптичні криві є важливим інструментом в сучасній криптографії. Вони надають ефективний та безпечний спосіб забезпечення автентичності, цілісності та конфіденційності даних. Використання еліптичних кривих дозволяє реалізувати безпечні криптографічні протоколи і забезпечує надійний захист інформації.

3 ЦИФРОВИЙ ПІДПИС НА RSA

Один з найраніших методів цифрового підпису являється цифровий підпис по схемі RSA. Ця схема була визначена Національним інститутом стандартів і технологій (NIST) в новому стандарті FIPS186-2 15 лютого 2000 року як стандарт цифрового підпису Сполучених Штатів. Вона є найпоширенішою схемою цифрового підпису у світі і вже розроблена в комерційне програмне забезпечення для використання бізнесом та приватними особами.

Схема цифрового підпису RSA базується на публічній криптосистемі RSA, запропонованій компанією, і базується на математичній складності розв'язання дискретних логарифмів над скінченим полем.

Загальний алгоритм цифрового підпису на RSA включає наступні кроки:

Генерація ключів:

1. Потрібно обрати два великих простих числа p і q.

2. Обчислити їх добуток $n = p * q$, який становить модуль RSA.

3. Обчислити значення функції Ойлера (Euler's totient function) $\varphi(n) = (p - 1) * (q - 1)$.

4. Вибрати ціле число e, яке є взаємно простим з $\varphi(n)$ і менше за $\varphi(n)$. Це стане публічним ключем (n, e).

5. Знайти число d, яке є оберненим до e модуля $\varphi(n)$, тобто $d * e \equiv 1 \pmod{\varphi(n)}$. Це стане приватним ключем (n, d).

Як здійснюєься підписування повідомлення:

1. Потрібно обчислити хеш повідомлення за допомогою хеш-функції.

2. Застосувати приватний ключ (n, d) до хешу, використовуючи піднесення до степеня за модулем n. Отримане значення є цифровим підписом.

Як здійснюється перевірка підпису:

1. Отримавши повідомлення, підпис та публічний ключ (n, e) потрібно застосувати публічний ключ до підпису, використовуючи піднесення до степеня за модулем n. Отримане значення, повинно збігатися з хешем повідомлення;

2. Обчисліть хеш отриманого повідомлення за допомогою тієї ж хеш-функції;

3. Порівняйте обчислений хеш з отриманим значенням під час перевірки підпису. Якщо вони збігаються, підпис вважається дійсним, інакше він вважається недійсним.

Важливо зауважити, що алгоритм RSA може використовувати різні хеш-функції, такі як SHA-256 або SHA-512, для обчислення хешу повідомлення. Крім того, в реальних реалізаціях RSA використовується додаткові оптимізації, такі як використання швидкого піднесення до степеня, але загальна сутність алгоритму залишається незмінною.

Переваги цифрового підпису RSA:

1. RSA вважається одним з найбільш безпечних алгоритмів цифрового підпису. Завдяки використанню великих простих чисел, що являється фактором складності факторизації, що в свою чергу робить його важким для зламування;

2. Цифровий підпис RSA забезпечує високу ступінь автентифікації. Він дозволяє перевірити, що повідомлення було підписано власником приватного ключа, а також підтвердити цілісність повідомлення;

3. RSA є швидким алгоритмом для перевірки цифрового підпису.

Недоліки цифрового підпису RSA:

1. RSA вимагає великого розміру ключа для забезпечення безпеки;

2. Підписування повідомлення використовуючи RSA може бути повільним у порівнянні з іншими алгоритмами цифрового підпису, особливо при використанні довгих ключів.

3. Хоча RSA вважається безпечним, він може бути вразливим до атак на факторизацію, якщо використовуються недостатньо великі прості числа p і q;

4. RSA не є стійким до квантових обчислень. З'явлення потужних квантових комп'ютерів може стати загрозою для безпеки RSA.

4 ЦИФРОВИЙ ПІДПИС ПО ELGAMAL

Схема цифрового підпису Ель-Гамаля була запропонована Т. Ель-Гамалем у 1985 році. Вона базується на проблемі дискретного логарифму і була розроблена переважно для цифрових підписів. Після RSA це є найвідомішою схемою цифрового підпису. З тих пір було багато поліпшень і розширень схеми Ель-Гамаля, наприклад, схема Харна, схема AMB, схема Йен-Лейн, схема ГОСТ та добре відома схема DSS/DSA, запропонована американським НБТ.

Цифровий підпис на Ель-Гамаль базується на математичних властивостях дискретного логарифму. Основні кроки для створення та перевірки цифрового підпису на Ель-Гамаль включають:

1. Генерація ключів:

- Потрібно обрати велике просте число p, яке відоме публічно, і згенерувати генератор g поля Галуа (або циклічну групу) порядку p;

- Вибрати випадкове число a, таке що $1 < a < p-1$, яке стане приватним ключем;

- Обчислити $b = g^a \bmod p$, яке стане публічним ключем.

2. Підписування повідомлення:

- Обчислити хеш повідомлення за допомогою хеш-функції;

- Обрати випадкове число k, таке що $1 < k < p-1$, яке є тимчасовим значенням;

- Обчислити $r = g^k \bmod p$;

- Обчислити s = (hash + a*r) * k^(-1) mod (p-1), де hash - хеш повідомлення.

3. Перевірка підпису:

- Отримати повідомлення, підпис r і публічний ключ (p, g, b);
- Обчислити хеш повідомлення за допомогою хеш-функції;
- Обчислити v = (g^hash * b^r mod p) mod p;
- Підпис вважається дійсним, якщо v = r.

Важливо враховувати, що при генерації ключів та виборі випадкових чисел необхідно дотримуватись правил безпеки для запобігання атакам, таким як повторне використання k або використання слабких простих чисел.

Переваги цифрового підпису на Ель-Гамаль:

1. Цифровий підпис на Ель-Гамаль є безпечним і заснованим на складних математичних проблемах, зокрема на проблемі дискретного логарифму;

2. Для алгоритма Ель-Гамаль не потрібна факторизація великих чисел, як у випадку RSA;

3. Ель-Гамаль має більшу криптографічну розмірність ключів порівняно з RSA при тому ж рівні безпеки;

4. В алгоритмі Ель-Гамаля існує можливість анонімного підпису, де підписувач може залишити анонімність, не розкриваючи свій приватний ключ або особисту інформацію.

Недоліки цифрового підпису на Ель-Гамаль:

1. Операції піднесення до степеня у цифровому підписі на Ель-Гамаль можуть бути обчислювально витратними, особливо при використанні великих простих чисел;

2. Підписи Ель-Гамала мають великий розмір порівняно з RSA;

3. Цифровий підпис на Ель-Гамаль може бути вразливий до атак, пов'язаних з витоком інформації про приватний ключ або використанням одного й того ж k для підписування різних повідомлень;

4. Розподіл публічних ключів у схемі Ель-Гамала може бути складним, особливо в порівнянні з RSA, де просто передаються публічні ключі. У Ель-Гамаля потрібен безпечний і надійний канал для обміну публічними ключами.

Це загальні переваги та недоліки цифрового підпису на Ель-Гамал. При виборі конкретного алгоритму підпису слід враховувати вимоги безпеки, продуктивності та конкретні потреби застосування.

5 ЦИФРОВИЙ ПІДПИС НА ЕЛІПТИЧНІЙ КРИВІЙ

Алгоритм Цифрового підпису (Digital Signature Algorithm, DSA) використовується в стандарті цифрового підпису (Digital Signature Standard, DSS). Початкова версія використовувала мультиплікативні групи скінченних полів. У більш новій версії з використанням еліптичних кривих (Elliptic Curve DSA, ECDSA) застосовується еліптична крива. Алгоритм є варіацією на схему підпису Ель-Гамаля з деякими змінами.

Опишемо основний алгоритм. Аліса хоче підписати документ m, який є цілим числом. Аліса обирає еліптичну криву над скінченним полем F_q так, що $\#E(F_q) = fr$, де r - велике просте число, а f - невелике ціле число, зазвичай 1, 2 або 4 (f має бути невеликим для забезпечення ефективності алгоритму). Вона обирає базову точку G в $E(F_q)$ порядку r. Нарешті, Аліса обирає секретне ціле число a і обчислює $Q = aG$. Аліса робить публічними наступні дані:

F_q, E, r, G, Q.

Для підпису повідомлення m Аліса виконує наступне:

1. Обирає випадкове ціле число k з умовою $1 \leq k < r$ і обчислює $R = kG = (x,y)$;

2. Обчислює $s = k^{-1}(m + ax) \pmod r$.

Підписаний документ має вигляд (m, R, s).

Для перевірки підпису Боб робить наступне:

1. Обчислює $u_1 = s^{-1}m \pmod r$ і $u_2 = s^{-1}x \pmod r$;

2. Обчислює $V = u_1G + u_2Q$;

3. Визначає підпис дійсним, якщо $V = R$.

Якщо повідомлення підписано правильно, то рівняння перевірки виконується:

$V = u_1G + u_2Q = s^{-1}mG + s^{-1}xQ = s^{-1}(mG + xaG) = kG = R$.

Основною відмінністю між ECDSA та системою Ель-Гамаля є процедура перевірки. У системі Ель-Гамаля рівняння перевірки

f(R)B + sR = mA вимагає трьох обчислень цілого числа на точку. Це є найбільш обчислювально витратною частиною алгоритму. У ECDSA потрібно лише два обчислення цілого числа на точку. Якщо планується багато перевірок, то покращена ефективність ECDSA є цінним фактором.

Розглянемо алгоритм для обчислення точок P ⊕ Q, [n]P, n=1,q

Припустимо, що у нас є еліптична крива з рівнянням $E:Y^2=X^3-15X+8$ з координатами точки P (7,16) та точки Q (1,2), як показано на малюнку нижче(рис. 5.1).

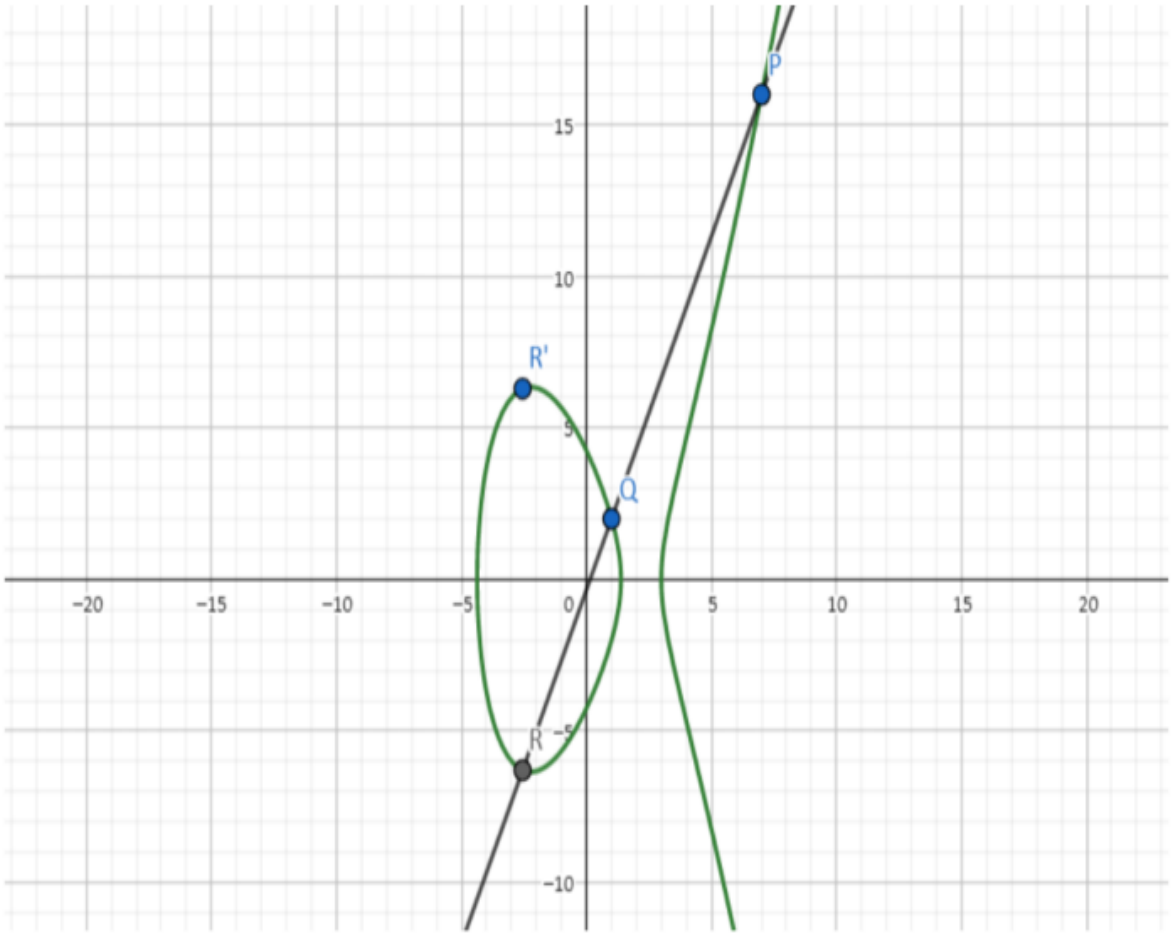


Рисунок 5.1 – Приклад еліптичної кривої з точкою P (7,16) та точки Q (1,2).

Спочатку нам потрібно отримати вираз для прямої PQ.

Очевидно, нахил прямої можна отримати як відношення різниці y-координат до різниці x-координат(5.1).

$$\lambda = \frac{X_P - X_Q}{Y_P - Y_Q} = \frac{7}{3} \tag{5.1}$$

Підставляючи координати точки P для отримання перетину, отримуємо вираз для прямої(5.2).

$$PQ: Y = \frac{7}{3}X - \frac{1}{3} \tag{5.2}$$

Замінюючи Y на лівій стороні рівняння E на Y тут, після спрощення, отримуємо(5.3).

$$X^3 - \frac{49}{9}X^2 - \frac{121}{9}X + \frac{161}{9} = 0 \tag{5.3}$$

Зауваживши, що пряма перетинає E в трьох точках, то це рівняння еквівалентне(5.4).

$$(x - e_1)(x - e_2)(x - e_3) = 0 \tag{5.4}$$

І ми вже знаємо, що два перетини прямої з E, P, Q відповідають двом розв'язкам x, тому ми маємо(5.5).

$$(x - 7)(x - 1)(x - e_3) = 0 \tag{5.5}$$

Спостерігаючи нульовий член двох еквівалентних рівнянь, маємо (5.6).

$$-7e_3 = \frac{169}{9} \rightarrow e_3 = \frac{23}{9} \tag{5.6}$$

Відомі координати на осі x, а ліва частина осі y відома після підстановки в рівняння. Тому отримуємо(5.7):

$$R = (-\frac{23}{9}, -\frac{170}{27}) \quad (5.7)$$

Як результат(5.8)*:*

$$\mathrm{P} \oplus \mathrm{Q} = R' = (-\frac{23}{9}, -\frac{170}{27}) \quad (5.8)$$

Проблема P $\oplus$ Q – вирішена.

Досвід в обчисленні говорить нам, що більш доцільно використовувати дотичну до E замість прямої PQ. Вона все ще задовольняє умову "пряма перетинає E в трьох точках", за винятком того, що дві точки співпадають в точці P. Це відображено в рівнянні, що обговорювалося раніше, яке є E" = P. На цій точці E# все ще добре вирішується.

6 ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗАДІЯНОГО У ПРОЕКТІ

При виконанні дипломної роботи, моєю головною задачею було написання алгоритма, та інтерфейса, які могли б наглядно продемонструвати виконання та перевірки цифрового підпису по Ель Гамалю з використанням еліптичної кривої. Для реалізації були обрані наступні інструменти:

- мова програмування JavaScript;
- середовище розробки Visual Studio Code;
- фреймворк для створення інтерфейсу React Bootstrap;
- середовище виконання програми – веб-браузер.

JavaScript має декілька переваг, які роблять його популярним вибором для веб-розробки та не тільки:

1. JavaScript розроблено для виконання в браузері, що робить його незамінним для створення динамічних та інтерактивних веб-сторінок;

2. JavaScript не обмежується тільки веб-розробкою. Він також використовується для створення серверних додатків за допомогою платформи Node.js, розробки мобільних додатків з використанням фреймворків, таких як React Native та NativeScript;

3. JavaScript являється мовою високого рівня через що вона має більш зрозумілий язик синтаксису для сприйняття і розуміння;

4. JavaScript є динамічною типізованою мовою, що дозволяє гнучко маніпулювати даними. Він не вимагає строгого визначення типів змінних, що дозволяє швидко експериментувати та розробляти прототипи.

Загалом, JavaScript є потужним і універсальним інструментом, який надає можливості для розробки різноманітних додатків та програм, забезпечуючи гнучкість, широку спільноту та велику кількість інструментів для полегшення роботи розробників.

Visual Studio Code (VS Code) - це безкоштовний і легкий у використанні текстовий редактор, розроблений компанією Microsoft. Він став

одним з найпопулярніших інструментів для розробки програмного забезпечення та веб-сайтів завдяки своїм функціональним можливостям, розширюваності та спрощеному інтерфейсу.

Основні особливості Visual Studio Code:

1. VS Code підтримує Windows, macOS та Linux, що робить його доступним для розробників на різних платформах;

2. Він має велику швидкість завантаження і реагування, що дозволяє розробникам працювати ефективно і без зайвих затримок;

3. VS Code надає масштабовану систему розширень, яка дозволяє розробникам налаштовувати та розширювати функціональність редактора;

4. VS Code має вбудовану підтримку системи керування версіями Git, що дозволяє зручно виконувати коміти, злиття гілок, переглядати зміни та історію проекту;

5. VS Code має вбудований дебаггер, який дозволяє розробникам крокувати по коду, встановлювати точки зупину та аналізувати значення змінних для виявлення та виправлення помилок;

6. Він надає вбудований термінал, що дозволяє виконувати команди та скрипти безпосередньо в редакторі;

7. VS Code підтримує роботу з командами та іншими розробниками, що дозволяє легко співпрацювати над проектами, використовуючи функції обміну посиланнями та відстеження змін.

Загалом, Visual Studio Code є потужним та розширюваним текстовим редактором, який надає розробникам зручність та продуктивність під час розробки програмного забезпечення та веб-додатків.

7 ПОВНИЙ РОЗБІР ПРОГРАМНОГО КОДУ ПРОЕКТА

Основною метою програмного проекта є створення та перевірка цифрового підпису на еліптичній кривій y2=x3-x+16 по алгоритма Ель Гамаля. Для досягнення цього була обрана базова точка G(5,7) за допомогою цієї базової точки і генеруються цифрові підписи. Також потрібно обрати розмір поля еліптичної кривої, в нашому випадку p=29 а її найбільший порядок n=31.

Знаючи основні параметри можна реалізовувати такі функції як:

- ініціалізація змінних;
- запис у змінні параметрів які обмежені через умову $1 \leq x \leq n-1$;
- функція яка вираховує значення R та s, що є основною інформацією цифрового підпису;
- функція яка обчислює V1 і V2 та порівнює їх, якщо вони збігаються - підпис вважається надійним, якщо ні то підпис підробляний.

Розгялнемо функції докладніше. Головні змінні коду цифрового підпису на еліптичній кривій за допомогою алгоритма Ель Гамаля, та їх опис продемонстровано в таблиці 7.1.

Таблиця 7.1 – Опис змінних в цифрового підпису на еліптичній кривій за допомогою алгоритма Ель Гамаля.

Назва	Опис
number1	Секретний ключ Аліси
number2	Відкритий ключ для Боба
hashSum	Хеш сума повідомлення
r	X координата точки перетину
s	Відома компонента підпису
publicKey	Відкритий ключ який ввів Боб
publicHash	Публічна Хеш сума яку ввів Боб
publics	Відома компонента підпису яку ввів Боб

Для ініціалізації змінних я обрав useState()(Лістинг 7.1)що є інструментом React, а саме функцією, яка приймає початкове значення стану і повертає масив з двох елементів: поточне значення стану та функцію для його оновлення.

Повний код ініціалізації цих змінних:

```
  //Інформація для Аліси
  const [number1, setNumber1] = useState();
  const [number2, setNumber2] = useState();
  const [hashSum, setHashSum] = useState();
  const [r,setR] = useState();
  const [s,setS] = useState();
//Інформація для Боба
  const [publicKey,setPublicKey] = useState();
  const [publicHash,setpublicHash] = useState();
  const [publics,setpublicS] = useState();
```

Лістинг 7.1 – Код ініціалізації змінних для роботи з алгоритмами

Коли змінні проініціалізовані в них потрібно записати певні параметри, це було виконано через input (Лістинг 7.2) поля, які записували значення в певну змінну при зміні числа в даному полі.

```
          <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="31"
onChange={handleNumber1Change} />
           <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="31"
onChange={handleNumber2Change} />
           <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="33"
onChange={handleHashSumChange} />
          <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="31"
onChange={handlePublicChange} />
      <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="31"
onChange={publicHashChange} />
           <input style={{ marginInlineStart: '20px',fontSize:
'18px'  }} type="number" min="1" max="33"
onChange={handleSChange} />
```

Лістинг 7.2 – Код input за допомогою яких записуються дані в змінні

Далі розглянемо функцію яка розраховує r – тобто координату точки перетину кривої за формулою: r= X_R mod n(Лістинг 7.3).

```
{(publicHash+(r*number1))%31}

setR(list[number2]);
```

Лістинг 7.3 – Код за допомогою якого знаходиться r

Тепер коли нам відомо значення r, ми можимо перейти до знаходження s – компонента підпису(Лістинг 7.4). Вона знаходиться за наступною формулою(7.1):

$$s = k^{-1}(e + k_A r) mod n$$

(7.1)

k-1- обернений елемент в кільці за модулем.

e- хеш-сумма повідомлення.

K_A- секретний ключ Аліси.

r – координата точки перетину кривої.

n- порядок кривої.

```
 for(let x = 1; x < 31; x++)
              if (((number2 % 31) * (x % 31)) % 31 === 1)
             setS((x*(hashSum+(number1*r)))%31)
             console.log("S",s);
    };
```

Лістинг 7.4 – Код знаходження s

Тепер коли ми знайшли усі потрібні невідомі, ми можемо створити цифровий підпис та передати відповідні параметри Бобу(Лістинг 7.5). Дані

які передаються: відкритий ключ, хеш сума повідомлення, та компонент підпису s.

```
       <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Відомий відкритий ключ: {number2}</p>
           <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Зашифроване повідомлення:</p>
           <p style={{ marginInlineStart: '20px',fontSize:
'13px'  }}>{encryptedMessage}</p>
     <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Відома компонента підпису s: {s}</p>
     <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Ключ для розшифрування: {secretSlovo}</p>
```

Лістинг 7.5- Передача відкритих даних на робочий стіл Боба

Тепер коли всі вище описані кроки виповнені ми можемо перейти до перевірки цифрового підпису. Після того як Аліса згенерелувала цифровий підпис, підписала ним повідомлення та передала Бобу всі потрібні дані, ми можемо виконати перевірку даного підпису, для цього виконуються дві формули (7.2)(7.3).

$$V1 = sR \tag{7.2}$$

$$V2 = h(M)G + rA \tag{7.3}$$

Потім результати порівнюються, і якщо вони дорівнюють одне одному, підпис вважається дійсним.

Програмна реалізація даного коду наведена в лістингу 7.6.

```
      <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Перевірка підпису:</p>
           <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>V1=sR mod 31={s}({r}G) mod 31 = {(s*number2)%31}G
</p><p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>V2=(h(M)G+rA) mod 31=({hashSum}G+{r}({number1}G)) mod
31= {(hashSum+(r*number1))%31}G</p>
           {((s*number2)%31)===((hashSum+(r*number1))%31)?
```

```
          <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Підпис вірний</p>:    <p></p>}
```

Лістинг 7.6 – код перевірки дійсності підписа

```
      <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Перевірка підпису:</p>
          <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>V1=sR mod 31={publics}({publicKey}G) mod 31 =
{(publics*publicKey)%31}G </p>
          <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>V2=(h(M)G+rA) mod 31=({publicHash}G+{r}({number1}G))
mod 31= {(publicHash+(r*number1))%31}G</p>
          {(publics*publicKey)%31===((publicHash+(r*number1))%31
)?
          <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Підпис дійсний</p>
         :
         <p style={{ marginInlineStart: '20px',fontSize:
'18px'  }}>Підпис не є дійсним</p>}
```

Лістинг 7.6, лист 2

На цьому робота алгоритма закінчюється. Також для зручності був написаний не складний інтерфейс користувача, який є інтуітивно зрозумілим і допомагає перевірити правильність роботи алгоритма, він симулює реальну перевірку цифрового підпису на певній еліптичній кривій за допомогою алгоритму Ель Гамаля, в реальному часі підставляються значення до формул, які користувач може бачити внизу екрану, також був розміщенний графік кривої та список с усіма можливими точками перетину кривої. Код для інтерфейсу наведений у лістингу 7.7.

На цьому етапі розбір програмного коду проекта закінчено.

8 ІНСТРУКЦІЯ ПО ВИКОРИСТАННЮ І ТЕСТУВАННЯ КОДУ

Після запуску програми користувач потрапляє на головне вікно програми, яке виглядає наступним чином див. рис. 8.1

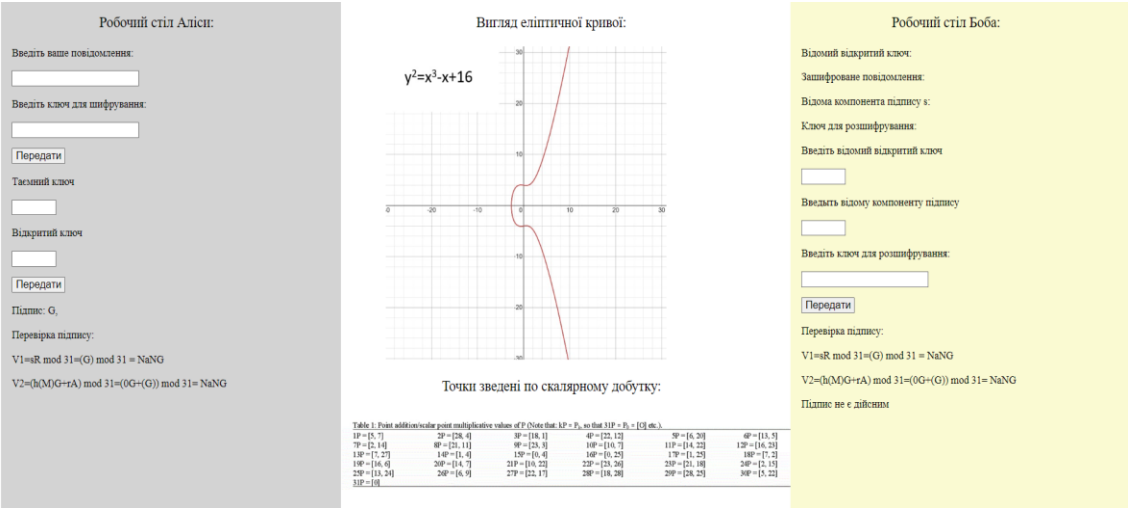


Рисунок 8.1 – Головний екран програми

Як бачимо інтерфейс програми поділений на три рівні частини, точніше стовпці, давайте розглянемо кожний стовпчик детальніше.

На першому стовпці (рис 8.2) ми бачимо «Робочий стіл Аліси:», як відомо Аліса створює цифровий підпис, отже тут ми повинні від її лиця ввести таємний ключ, відкритий ключ повідомлення та ключ розшифровки для Боба. Ці данні потрібно вводити у відповідні поля, потім коли ми ввели останнє значення, натискаємо кнопку «Передати». Все це потрібно для створення цифрового підпису на еліптичній кривій y2=x3-x+16 по алгоритму Ель Гамаля з базовою точкою G(5,7).

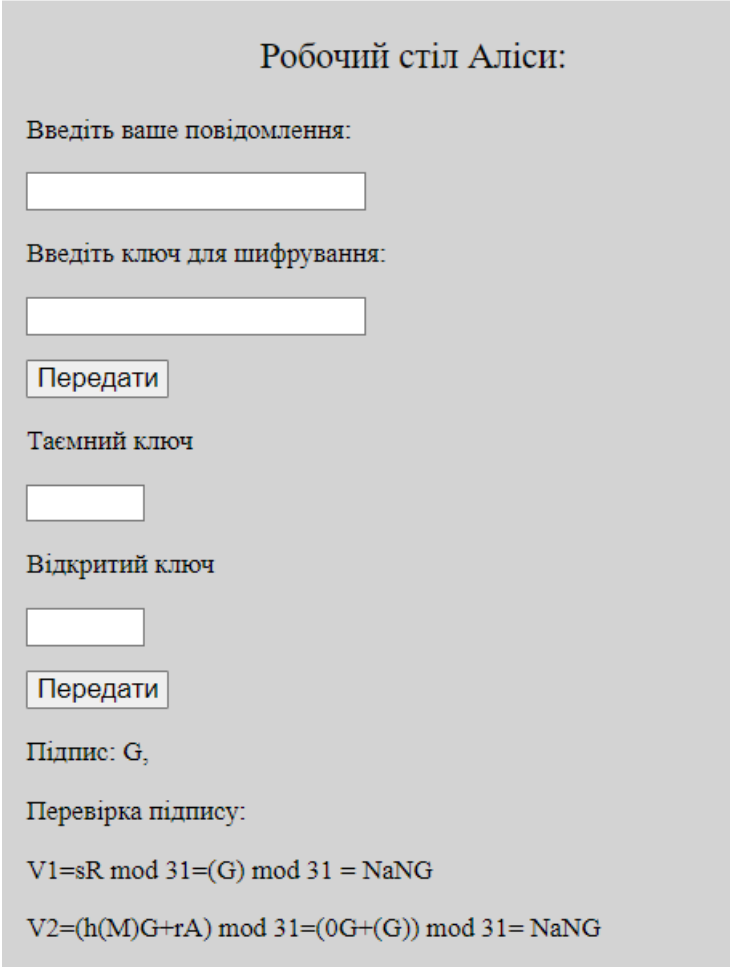
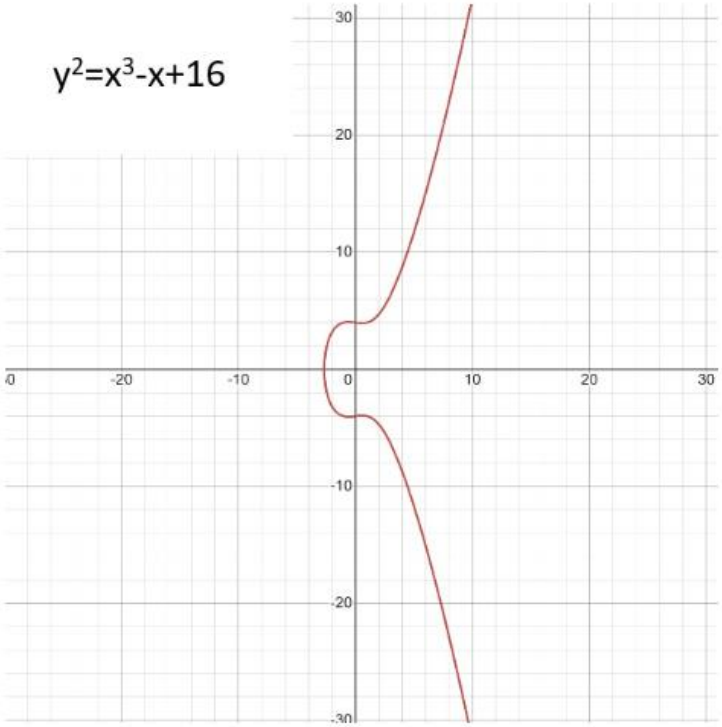


Рисунок 8.2 – Робочий стіл Аліси

В наступному стовпці можемо побачити наступну інформацію(рис. 8.3):

- Вигляд еліптичної кривої;
- Формулу еліптичної кривої;
- Точки зведені по скалярному добутку.

Вигляд еліптичної кривої:



Точки зведені по скалярному добутку:

Table 1: Point addition/scalar point multiplicative values of P (Note that: $kP = P_k$, so that $31P = P_0 = [O]$ etc.).

1P = [5, 7]	2P = [28, 4]	3P = [18, 1]	4P = [22, 12]	5P = [6, 20]	6P = [13, 5]
7P = [2, 14]	8P = [21, 11]	9P = [23, 3]	10P = [10, 7]	11P = [14, 22]	12P = [16, 23]
13P = [7, 27]	14P = [1, 4]	15P = [0, 4]	16P = [0, 25]	17P = [1, 25]	18P = [7, 2]
19P = [16, 6]	20P = [14, 7]	21P = [10, 22]	22P = [23, 26]	23P = [21, 18]	24P = [2, 15]
25P = [13, 24]	26P = [6, 9]	27P = [22, 17]	28P = [18, 28]	29P = [28, 25]	30P = [5, 22]
31P = [0]					

Рисунок 8.3 – Інформація про еліптичну криву та точки зведені по скалярному добутку

Останній стовпчик відповідає за «Робочий стіл Боба:»(рис8.4).

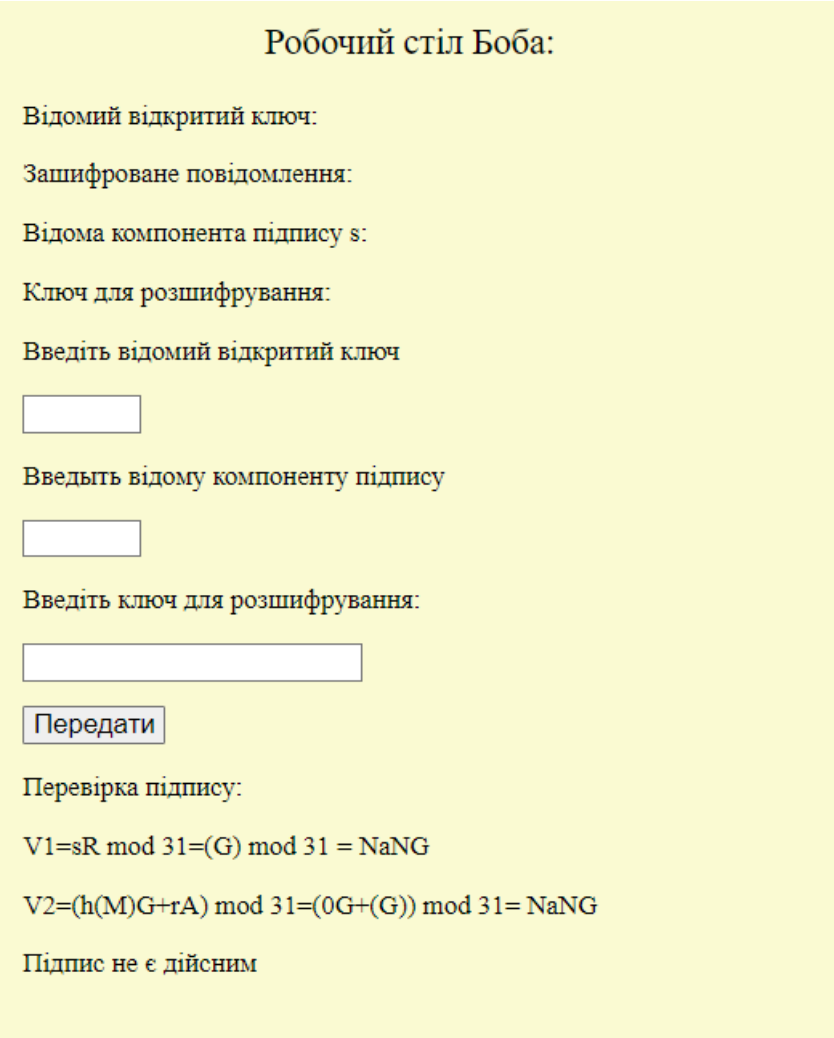


Рисунок 8.4 – Робочий стіл Боба

Якщо Аліса вже заповнила необхідні дані для цифрового підпису в полях «Відомий відкритий ключ:», «Зашифроване повідомлення», «Відома компонента підпису s:», «Ключ для розшифрування» будуть дані які вписала Аліса (рис. 8.5), в іншому випадку вони будуть пустими. В Боба також є три поля, він може ввести данні які йому відомі, або ж для перевірки алгоритму навмисне вписати в якесь поле хибні данні, в такому випадку він отримає таку відповідь (рис. 8.6).

Робочий стіл Боба:

Відомий відкритий ключ: 12

Зашифроване повідомлення:

U2FsdGVkX19eN0KFRANs4OppHvWXL0nTt3wZSW7DhU0ZIopli3UrUBy8ZEj5fa8E

Відома компонента підпису s: 29

Ключ для розшифрування: 23да3

Рисунок 8.5 – Заповнені поля з інформацією від Аліси

Робочий стіл Боба:

Відомий відкритий ключ: 12

Зашифроване повідомлення:

U2FsdGVkX19eN0KFRANs4OppHvWXL0nTt3wZSW7DhU0ZIopli3UrUBy8ZEj5fa8E

Відома компонента підпису s: 29

Ключ для розшифрування: 23да3

Введіть відомий відкритий ключ

13

Введыть відому компоненту підпису

29

Введіть ключ для розшифрування:

23да

Передати

Перевірка підпису:

V1=sR mod 31=29(13G) mod 31 = 5G

V2=(h(M)G+rA) mod 31=(16G+16(13G)) mod 31= 7G

Розшифроване повідомлення: Неправильний ключ

Підпис не є дійсним

Рисунок 8.6 – Боб навмисне вписав неправильні дані

Тепер давайте розглянемо повну роботу програми. Спочатку заповнюємо в першому стовпчику поля з повідомленням та ключем шифрування і натискаємо «Передати», потім заповнюємо поля з таємним та відкритим ключем, потім натискаємо «Передати» та бачимо, що підпис згенеровано правильно(рис. 8.7).

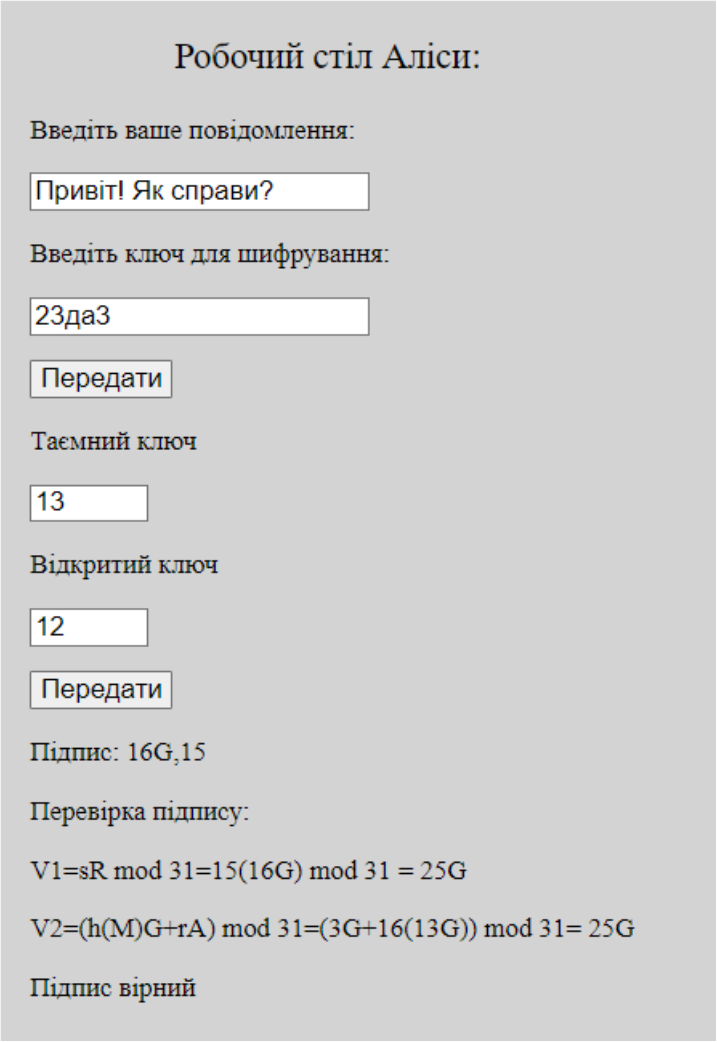


Рисунок 8.7 – Генерація підпису Алісою

Як вже було показано на рисунку 8.6 Боб отримав певні дані, але в тому випадку він навмисне ввів дані неправильно, на цей раз подивимося що буде, якщо Боб введе всі дані правильно(рис. 8.8).

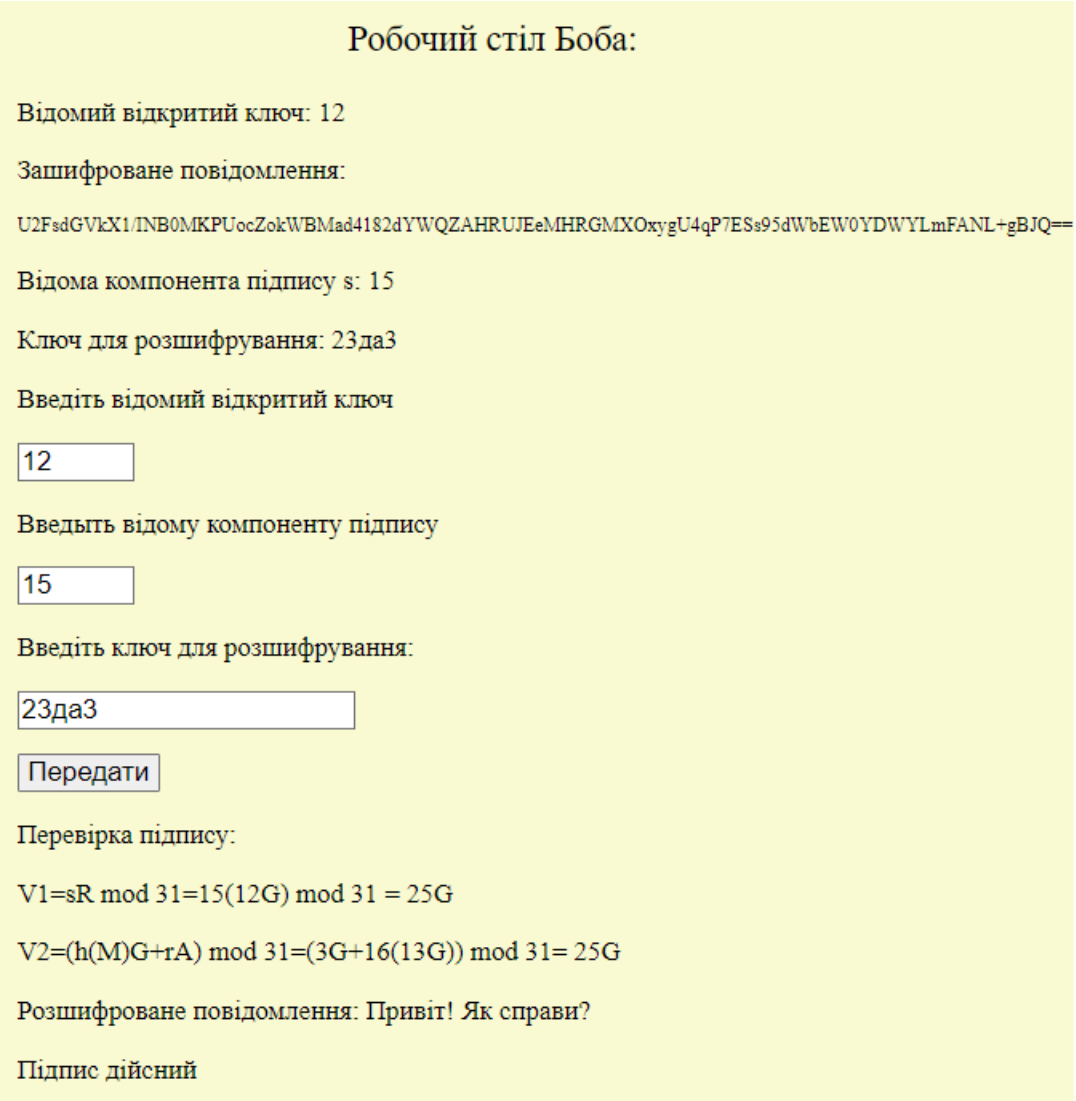


Рисунок 8.8 – Перевірка на правильний цифровий підпис

Як бачимо під кнопкою «Передати» на робочому столі Боба проходить перевірка цифрового підпису за формулами, які були описані в 7 пункті, а саме формули (7.2) та (7.3). У ці формули підставляються данні які ввів Боб та проводиться рахування, якщо кінцевий результат збігається, підпис вважається дійсним, такому повідомленню можна довіряти.

ВИСНОВОК

У результаті виконання дипломної роботи, було детально розібрано дослідження алгоритмів цифрового підпису на еліптичних кривих, таких як алгоритми ECDSA(Elliptic Curve Digital Signature Algorithm), та ElGamal.

У процесі роботи були досліджені основні математичні принципи еліптичних кривих, а також розглянуті принципи функціонування алгоритму, та його застосування для цифрових підписів.

Також була виконана розробка власного алгоритму цифрового підпису на еліптичних кривих з урахуванням сучасних вимог безпеки та ефективності. У мого алгоритму є багато способів удосконалення, наприклад:

- Надати змогу обирати свою еліптичну криву;
- Замість того щоб одразу вписувати хеш-суму повідомлення, написати функцію яка по певному алгоритму сама б рахувала хеш-суму вписаного користувачем повідомлення;
- Замість статичної картинки кривої, можна було розробити програмний код, який би в реальному часі з наданих параметрів будував еліптичну криву, точки та прямі на цій еліптичній кривій.

Результати роботи можуть мати практичне застосування у сферах, які потребують надійного забезпечення цифрових підписів, такі як: електронна комерція, фінансові транзакції та безпека інформаційних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kefa Rabah, Elliptic Curve ElGamal Encryption and Signature Schemes. [Електронний ресурс] – Режим доступу: https://www.researchgate.net/publication/45949429_Elliptic_Curve_ElGamal_Encryption_and_Signature_Schemes.

2. LAWRENCE C. WASHINGTON, Elliptic Curves Number Theory and Cryptography Second Edition. [Електронний ресурс] – Режим доступу: https://people.cs.nctu.edu.tw/~rjchen/ECC2012S/Elliptic%20Curves%20Number%20Theory%20And%20Cryptography%202n.pdf.

3. Tim G¨uneysu, Christof Paar, Jan Pelzl, On the Security of Elliptic Curve Cryptosystems against Attacks with Special-Purpose Hardware. [Електронний ресурс] – Режим доступу: https://www.researchgate.net/publication/228589576_On_the_security_of_elliptic_curve_cryptosystems_against_attacks_with_special-purpose_hardware.

4. Ian F. Blake, Gadriel Seroussi, Nigel P. Smart Advances in Elliptic Curve Cryptography. [Електронний ресурс] – Режим доступу: https://cdn.preterhuman.net/texts/cryptography/Cambridge%20University%20Press.%20Advances%20in%20Elliptic%20Curve%20Cryptography%20(2005).pdf.

5. Marcel Medwed, Elisabeth Oswald, Template Attacks on ECDSA. [Електронний ресурс] – Режим доступу: https://www.researchgate.net/publication/221239700_Template_Attacks_on_ECDSA.