МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

УДК 004.71

До захисту допускається
Т.в.о завідувача кафедри
комп'ютерних наук та інженерії
к.т.н, доц. Сафонова С. О.
_____
«____»_____ 2020 р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

«Дослідження алгоритму створення електронного цифрового підпису з використанням групи точок еліптичної кривої»

Освітньо-кваліфікаційний рівень «Магістр»
Спеціальність 122 «Комп'ютерні науки»

Науковий керівник роботи: _____ Кардашук В. С.
(підпис) (ініціали, прізвище)

Консультант з охорони праці: _____ Критська Я. О.
(підпис) (ініціали, прізвище)

Студент: _____ Сандулов В. Ю.
(підпис) (ініціали, прізвище)

Група: КН-18 дм

Сєвєродонецьк – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет інформаційних технологій та електроніки
Кафедра комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Спеціальність 122 «Комп'ютерні науки»

«**ЗАТВЕРДЖУЮ**»
Т.в.о завідувача кафедри
комп'ютерних наук та інженерії
к.т.н, доц. Сафонова С. О.

"____" _____ 2020 року

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Сандулову Владиславу Юрійовичу

(прізвище, ім'я, по-батькові)

1. **Тема проекту (роботи):** «Дослідження алгоритму створення електронного цифрового підпису з використанням групи точок еліптичної кривої» затверджена наказом по університету № 135/15.15 від «11» жовтня 2019 р.

2. Строк здачі студентом закінченого проекту (роботи): 10.01.2020 р.

3. Вихідні дані проекту (роботи): матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити):

1. Дослідити та розробити алгоритми виконання операцій нижнього рівня на еліптичній кривій;
2. Розробити програмне забезпечення для виконання алгоритму електронно-цифрового підпису на еліптичних кривих;
3. Розробити програмне забезпечення для реалізації методів багатократного множення точок еліптичної кривої на число;
4. Запропонувати метод високошвидкісного множення точок еліптичної кривої на число та провести аналіз швидкодії;
5. Охорона праці та безпека в надзвичайних ситуаціях.

5. Перелік графічного матеріалу (з точною назвою обов'язкових креслень):

не передбачено

6. Консультанти роботи, з вказівкою розділів, що до них відносяться

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основна частина	Кардашук В. С.		
Охорона праці та безпека в надзвичайних ситуаціях	Критська Я. О.		

7. Дата видачі завдання _____

Керівник _____ Кардашук В. С.

(підпис)

Завдання до виконання прийняв _____ Сандулов В. Ю.

(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітки
1.	Отримання завдання, збір матеріалів	18.10.19- 24.10.19	
2.	Огляд літератури й обґрунтування необхідності дослідження	25.10.19 –28.10.19	
3.	Дослідження методів та засобів криптографічного перетворення на основі точок еліптичної кривої	29.10.19 – 28.11.19	
4.	Розроблення алгоритмів реалізації операцій над точками еліптичної кривої	28.11.19 –31.12.19	
5.	Розроблення програмного забезпечення	03.01.20 – 04.01.20	
6.	Оформлення пояснювальної записки	05.01.20 – 08.01.20	
7.	Підготовка та подання магістерської роботи до захисту	09.01.20 – 10.01.20	

Студент _____

(підпис)

Науковий керівник _____

(підпис)

АНОТАЦІЯ

Сандулов В. Ю. Дослідження алгоритму створення електронного цифрового підпису з використанням групи точок еліптичної кривої

В роботі розглянуто математичні основи та принципи еліптичної криптографії, аналіз методів шифрування, що використовують еліптичну криптографію, огляд алгоритмів генерування та перевірки електронно-цифрового підпису з використанням еліптичної криптографії, принципи реалізації операцій над точками еліптичної кривої, аналізуються алгоритми додавання та множення точок еліптичної кривої на число, розглядаються алгебраїчні структури, що використовуються для визначення параметрів точки еліптичної кривої та їх особливості, а також узагальнено описуються принципи реалізації дій над точками еліптичної кривої в електронно-обчислювальних пристроях.

Розроблено архітектуру програмного забезпечення для реалізації алгоритму електронно-цифрового підпису. Обґрунтовуються принципи та підходи, покладені в основу програмного комплексу для реалізації алгоритму електронно-цифрового підпису з використанням еліптичних кривих.

Ключові слова: асиметрична криптографія, еліптична крива, цифровий підпис, ECDSA, множення точок еліптичної кривої**.**

ABSTRACT

Sandulov V. Y. Investigation of algorithm of creation of electronic digital signature using group of points of elliptic curve

The mathematical foundations and principles of elliptic cryptography, analysis of encryption methods using elliptical cryptography, review of algorithms for generating and verification of electronic-digital signature using elliptical cryptography, principles of operations on points of elliptic curve, analysis of elliptical points are analyzed number, the algebraic structures used to determine the parameters of the point of the elliptic curve and their peculiarities, as well as the general describes the principles of the action points on an elliptic curve in electronic computing devices.

Software architecture for implementation of electronic-digital signature algorithm has been developed. The principles and approaches that underpin the software complex for the implementation of the electronic-digital signature algorithm using elliptic curves are substantiated.

Keywords: asymmetric cryptography, elliptic curve, digital signature, ECDSA, multiplication of the points of the elliptic curve.

ЗМІСТ

ВСТУП

З бурхливим розвитком галузі електронного документообігу, питання захисту та забезпечення достовірності даних та документів, що передаються незахищеними каналами зв'язку, такими як мережа Інтернет, займає чільне місце серед досліджень, що проводяться в даній галузі. Розроблений в даній роботі модифікований метод багатократного скалярного множення точок еліптичної кривої на число має особливу цінність в контексті зменшення часу на перевірку електронно-цифрового підпису, що в свою чергу впливає на загальну швидкодію алгоритму [1].

Однією з галузей застосування методів електронно-цифрового підпису, що має практичний інтерес та потенціал до адаптації запропонованих в межах даної роботи розробок є сфера державного електронного документообігу.

Основна проблема існуючого стану речей в даній сфері полягає в низькому ступені проникнення алгоритмів, заснованих на еліптичній криптографії в сфери державного управління. На даний час, державні стандарти використання еліптичної криптографії хоч і існують, але поки що не набули широкого використання на рівні державних структур, адже основні операції електронно-цифрового підпису, як наприклад, використання ЕЦП (електронного цифрового підпису) в регуляції діяльності фізичних осіб підприємців Державною Фіскальною Службою України засноване на використанні алгоритму RSA (абревіатура від прізвищ Rivest, Shamir та Adleman — криптографічний алгоритм з відкритим ключем), що має значно нижчі показники швидкодії у порівнянні з алгоритмом ECDSA (Elliptic Curve Digital Signature Algorithm – алгоритм з відкритим ключем для створення цифрового підпису на основі еліптичних кривих).

Виходячи з вищеописаної інформації, можна виділити ряд конкретних недоліків притаманних ЕЦП на основі алгоритму RSA.

По-перше це низька криптостійкість електронно-цифрового підпису з визначеною довжиною в порівнянні з підписом аналогічного розміру з використанням алгоритму ECDSA. Це означає що підпис виконаний з використанням принципів еліптичної криптографії мав би набагато вищу криптостійкість.

По-друге, нижча швидкодія алгоритму RSA в порівнянні із запропонованим модифікованим алгоритмом ECDSA означає, що за певний виділений проміжок часу, розробник документів може виконати меншу кількість операцій, що призводить до уповільнення процесу загалом.

По-третє, навіть припускаючи використання алгоритму ECDSA з існуючими методами багатократного скалярного множення точок еліптичної кривої на число,

необхідний великий об'єм пам'яті для зберігання масиву передобчислених значень. Виходячи з цього, існує нагальна потреба скорочення об'ємів апаратних ресурсів, що використовуються в процесі електронно-цифрового підпису.

В сучасному світі важливість захисту інформації має надзвичайно важливу роль. З розвитком інформаційних технологій, кількість даних, що використовується в процесі людської діяльності нестримно зростає. Даний процес веде до збільшення ризиків, пов’язаних з витоками чутливої інформації та використання її третьою стороною в шкідливих цілях. В даному контексті слід згадати процес інформатизації сфер діяльності, що оперують персональними даними такими як: біометричні дані, банківські реквізити, відомості що відносяться до сфери приватного життя тощо. Також ілюстрацією даної проблематики, що говорить сама за себе, є проблема засвідчення оригінальності, незмінності та приватності даних, що передаються через відкриті канали такі як мережа Інтернет [2].

Для засвідчення оригінальності документу або пакету даних, що передається між відправником, що гарантує свою достовірність та отримувачем, що бажає пересвідчитись у факті незмінності отриманої інформації в процесі передачі незахищеними каналами, використовується механізм електронно-цифрового підпису. Даний механізм є за своєю суттю аналогічним до процесу фізичного підписання документу, коли укладачі документу своїм підписом підтверджують свою особистість.

Існує безліч підходів, що дозволяють реалізувати механізм електронно-цифрового підпису. В межах даної роботи особлива увага приділяється методам, заснованим на теоріях алгебраїчної геометрії, зокрема теорії еліптичних кривих.

Використання еліптичних кривих для створення криптосистем було незалежно запропоновано Нілом Коблицем та Віктором Міллером у 1985 році. Ключова перевага даної методології ґрунтується на двох основних особливостях точок еліптичної кривої: розподіл результату додавання точок еліптичної кривої є у великій мірі рівномірним, тобто покриває всю область значень даної кривої; наслідком цієї особливості є проблема дискретного логарифмування для еліптичних кривих (ECDLP). Ці особливості обумовлюють високу криптографічну стійкість систем з використанням еліптичних кривих.

Слід відмітити, що ще у 2002 році прийнято ДСТУ 4145-2002 «Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих» [3].

Актуальність теми: На сьогоднішній день галузь шифрування та захисту інформації відіграє важливу роль в царині інформатики. Це зумовлено масштабним використанням автоматизованих методів обробки та передачі даних та широким

розповсюдженням методів та засобів несанкціонованого доступу до інформації, що пересилається по незахищеними каналами зв'язку.

Проблему передачі деякої конфіденційної інформації адресату можна вирішити багатьма способами, проте найбільш часто використовуваними в наш час є так звані асиметричні криптосистеми. Низка провідних досліджень показує, що серед асиметричних криптосистем, або криптосистем з відкритим ключем найбільш стабільною є криптосистема з використанням еліптичних кривих. Дані криптосистеми для забезпечення достатньої криптографічної стійкості потребують довжини ключа в кілька разів меншу за ту, що необхідна для криптосистем на основі математичних операцій в скінченних полях.

Одним з яскравих представників сімейства криптографічних алгоритмів з використанням еліптичних кривих є алгоритм електронно-цифрового підпису ECDSA [4].

При застосуванні криптографічних алгоритмів з використанням еліптичних кривих дуже важливим чинником є час їхньої роботи. Експериментальні дослідження показують, що найбільш ресурсо- та часовитратними є операції, що виконуються безпосередньо з точками еліптичної кривої, зокрема для алгоритму ECDSA найбільш ресурсовитратною є операція багатократного скалярного множення точок еліптичної кривої на число. Таким чином, актуальними є дослідження способів та методів оптимізації даного обчислення.

Об'єкт дослідження: процеси обміну ключами, шифрування, дешифрування та створення/перевірка електронно-цифрового підпису в еліптичних криптосистемах.

Предмет дослідження: методи багатократного скалярного множення точок еліптичної кривої у полі GF(p) та $GF(2^m)$.

Методи дослідження: методи теорії чисел, аналітичної геометрії, методи абстрактної алгебри, дискретної математики та криптографії.

Мета дослідження: розроблення та дослідження методу високошвидкісного множення точок еліптичної кривої на число.

Наукова новизна полягає в наступному:

Запропоновано модифікацію методу багатократного скалярного множення точки еліптичної кривої на число, яка полягає у виконанні спеціального передобчислення, і на відміну від існуючих методів дозволяє звести виконання операції множення точки еліптичної кривої на число до додавання, що забезпечує вищу швидкодію (до 25%) порівняно з існуючими для еліптичних кривих з параметрами над GF(p) та $GF(2^m)$.

Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму багатократного скалярного множення точки еліптичної кривої на число, що

дозволяє на основі шаблону замінювати конкретну реалізацію в екземплярі класу електронно-цифрового підпису без застосування наслідування.

Апробація результатів роботи. Основні результати роботи представлені у наступних публікаціях:

1. Сандулов В.Ю. Створення електронного цифрового підпису з використанням групи точок еліптичної кривої / Матеріали IX Всеукраїнської науково-практичної конференції “Електроніка та телекомунікації” (8-9 листопада 2019 р.). – Сєвєродонецьк. – С. 146-148.

2. Сандулов В.Ю. Сучасний підхід до формування криптосистем на базі еліптичних кривих / Матеріали Всеукраїнської науково-практичної конференції з міжнародною участю «Майбутній науковець-2019» (12 грудня 2019 р.). – Сєвєродонецьк. – С. 182-184.

Практичне використання результатів полягає у наступному: запропоновані алгоритми дозволяють швидше виконувати операцію багатократного скалярного множення точок еліптичної кривої на число, що дозволяє скоротити час виконання алгоритму електронно-цифрового підпису.

Розроблене математичне та програмне забезпечення для виконання операцій з точками еліптичної кривої дозволяє спростити подальші дослідження у галузі еліптичної криптографії.

Розроблений програмний комплекс може бути використаний в навчальному процесі кафедри комп’ютерних наук та інженерії при проведені практичних занять з дисципліни «Захист інформації в комп’ютерних системах».

Структура і обсяг роботи.

Магістерська робота складається зі вступу, 4 розділів, висновків, переліку посилань до розділів з 44 найменувань, додатку на 26 сторінках. Загальний обсяг роботи складає 114 сторінок. Магістерська робота містить 24 рисунки та 4 таблиці.

РОЗДІЛ 1
АНАЛІЗ КРИПТОСИСТЕМ НА ОСНОВІ ЕЛІПТИЧНИХ КРИВИХ

1.1. Аналіз сучасних підходів до формування криптосистем на базі еліптичних кривих

Розвиток інформаційних технологій призвів до виникнення окремого розділу криптографії – блочного шифрування. Сьогодні все більшу популярність набирають методи криптографії, засновані на еліптичних кривих, але не адаптованість цих методів до захисту інформації не дає можливості використовувати їх у сучасних каналах зв'язку [5].

Криптографія на еліптичних кривих — напрям асиметричного шифрування даних, що швидко розвивається з використанням сучасних інформаційних технологій. У криптографії на еліптичних кривих усі обчислення (наприклад, вибір значення ключа) проводяться над точками еліптичної кривої, тобто, замість звичайного складання двох чисел виконується за певними правилами складання двох точок кривої, при цьому як результат виходить третя точка.

Цифровий підпис файлів або електронних поштових повідомлень виконується з використанням криптографічних алгоритмів, що використовують несиметричні ключі. Власне для підпису використовується секретний ключ, а для перевірки чужого підпису відкритий. Ключі є числами довжини від 512 до 4096 біт математично або функціонально пов'язаними між собою.

Криптографічним алгоритмом, що стандартно використовується для методів шифрування симетричними ключами (для цілей поширення) є RSA (Rivest, Shamir і Adleman). Хоча RSA має високу міру захисту і широко застосовується, його застосування пов'язане з деякими проблемами та питанням криптостійкості при сучасному розвитку технологій. Альтернативна технологія криптографії на еліптичних кривих, заснована на математичному методі використання функції еліптичних кривих та дає істотні переваги перед RSA [6].

В алгоритмах цифрового підпису активно використовуються обчислення в кінцевих полях Галуа. Ціле позитивне число a порівняно з b за модулем $p(a \equiv b \bmod p)$, якщо залишок від ділення b на p дорівнює a.

Можна ввести операції складання і множення за модулем p. Результатом складання двох чисел за модулем p, вважатиметься залишок від ділення їх суми на число p. Неважко помітити, що результати операцій складання або множення пари довільних ненегативних цілих чисел за модулем p не перевершуватимуть число p. У результаті, можна обмежиться

розглядом безлічі чисел *0,1...p – 1* із заданими на них операціями складання і множення за модулем *p*. Множина *0,1...p – 1* із заданими операціями складання і множення, що підкоряються звичайним законам складання, множення і розкриття дужок, утворюють кільце класів розрахунків за модулем p. Елемент b називається зворотним до елементу *a*, якщо *ab = 1*. Зворотній елемент позначається *a–1*. Оперуючи тільки цілими ненегативними числами, неважко ввести операцію ділення як множення на зворотний елемент, операцію віднімання і навіть негативні числа. Виявляється, якщо p — просте число, то зворотний елемент існує для усіх елементів кільця (окрім природно числа O).

Еліптичною кривою називають безліч пар точок (X, Y), у таких що задовольняють рівнянню [7]:

$$y^2 = x^3 + ax + b \tag{1.1}$$

Можна накласти обмеження на безліч значень змінних x, y і коефіцієнтів a, b. Обмежуючи область визначення рівняння значущою для застосувань числовою множиною ми отримаємо еліптичну криву, задану над даним полем.

У додатку до ДСТУ 4145-2002 еліптична крива над кінцевим простим полем GF(p) визначається як безліч пар *(x, y),* таких що *x, y ≡ GF (p)* та задовольняють рівнянню:

$$y^2 = x^3 + ax + b \bmod p; \quad a, b \equiv GF(p) \tag{1.2}$$

Пари (x, y) є точками еліптичної кривої. Точки еліптичної кривої можна складати. Сума двох точок, у свою чергу, теж лежить на еліптичній кривій.

Математична властивість, яка робить еліптичні криві корисними для криптографії, полягає в тому, що якщо взяти дві різні точки на кривій, то хорда, що сполучає їх, перетне криву в третій точці (оскільки ми маємо кубічну криву). Дзеркально відбивши цю точку по осі X, отримуємо ще одну точку на кривій (оскільки крива симетрична відносно осі X). Якщо позначити дві первинні точки як P і Q, то отримаємо останню — відбиту точку P + Q (рис. 1.1).

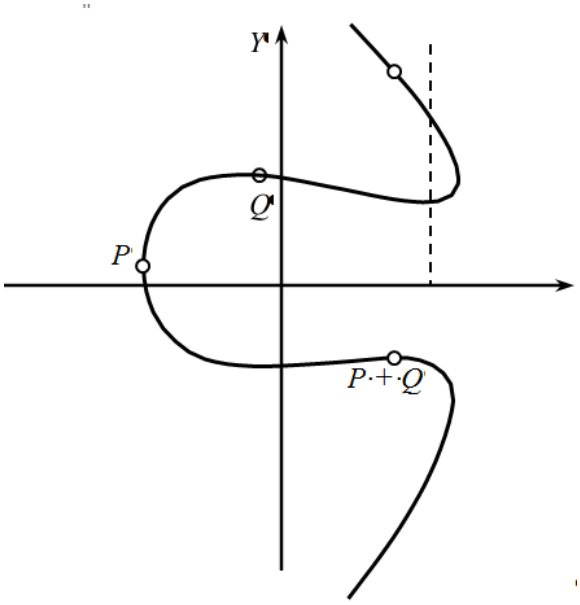


Рисунок 1.1 – Додавання точок на еліптичній кривій

Це складання задовольняє всім відомим правилам алгебри для цілих чисел.

Окрім точок, що лежать на еліптичній кривій, розглядається також нульова точка. Вважається, що сума двох точок — A з координатами (X_A, Y_A) і B з координатами (X_B, Y_B) — рівна нулю, якщо $X_A = X_B$, $Y_A = -Y_B$ (mod p). Нульова точка не лежить на еліптичній кривій, але, проте, бере участь в обчисленнях. Її можна розглядати як нескінченно видалену точку [8].

Таким чином, можна визначити кінцеву абелеву групу точок кривої, де нулем буде нескінченно видалена точка. Зокрема, якщо точки P і Q збігаються, то можна вичислити P + P, тобто 2P. Розвиваючи цю ідею, можна визначити kP для будь-якого цілого числа k, і отже, визначити значення P і значення найменшого цілого числа k, такого, що kP = F, де F — нескінченно віддалена точка.

Тепер можна сформулювати проблему дискретного логарифма еліптичної кривої, на якій заснована дана система: «базова точка P і розташована на кривій точка kP; знайти значення k».

1.2 Методи шифрування в еліптичній криптографії

Найбільш популярними напрямками еліптичної криптографії, тобто сферами, в яких криптостійкість базується на задачі дискретного логарифмування для еліптичних кривих

або ECDLP, є шифрування з відкритим ключем та алгоритм електронно-цифрового підпису.

Існують дві схеми обміну ключами [9]:

– симетрична – існує єдиний секретний ключ, що має бути переданий між відправником та отримувачем захищеним каналом;

– асиметрична – існує пара ключів – приватний та публічний, що може бути переданий незахищеним каналом).

В межах даної роботи детально розглядається варіант асиметричного шифрування.

1.2.1 Еліптичний варіант обміну ключами по алгоритму Діффі-Хелмана

Алгоритм Діффі-Хелмана – це асиметричний метод обміну криптографічними ключами. Даний метод дозволяє двом учасникам, що не мають жодних попередніх даних один про одного, отримати спільний секретний ключ, що використовуватиметься для шифрування даних, якими обмінюються сторони, за допомогою незахищеного каналу зв'язку. Цей ключ можна використати для шифрування наступних сеансів зв'язку, що використовують шифр з симетричним ключем [10].

Нехай існує еліптична крива, що забезпечує достатню криптостійкість (несуперсингулярну, в якій генеруюча точка G = (x; y) має великий порядок, тобто число n, при якому nG = O є дуже великим простим числом), визначена параметрами a та b : $y^2 = x^3 + ax + b.$

Позначимо сторону відправника як A, сторону отримувача як B, тоді обмін ключами між сторонами A та B проводиться таким чином:

1. Сторона A обирає ціле число PrivA < n . Дане число є приватним ключем учасника, а точка еліптичної кривої $Pub_A = Priv_A \cdot G$ є публічним ключем.

2. Сторона B обирає аналогічно секретний ключ $Priv_B$ та обчислює відкритий ключ $Pub_B = Priv_B \cdot G$.

3. Учасник A генерує секретний ключ $K = k_A \cdot P_B$, а учасник B генерує секретний ключ $K = k_B \cdot P_A$.

Дві формули, отримані в п.3 дають один й той самий результат, оскільки:

$$\kappa A \times \Pi B = \kappa A \times (\kappa B \times \Gamma) = \kappa B \times (\kappa A \times \Gamma) = \kappa B \times \Pi A \qquad (1.3)$$

Проблема, що стоїть перед сторонніми спостерігачами, які мають намір дізнатись секретний ключ, полягає в обчисленні $kA \times kB \times G$ за відомими G, $kA \times G$, $kB \times G$, але не знаючи kA та kB . Це і є проблемою Діффі-Хеллмана для еліптичних кривих.

1.2.2 Протокол Мессі-Омура (Massey-Omura)

Криптосистема Мессі-Омура [4] була запропонована в 1978 році Джеймсом Мессі і Джимом К. Омура в якості поліпшення протоколу Шаміра. Є два варіанти реалізації даного протоколу: класичний і еліптичний. Перший побудований на складності завдання дискретного логарифмування, другий на властивостях еліптичної кривої.

Еліптичний варіант даного протоколу надає можливість передавати повідомлення від відправника А до отримувача В по відкритому каналу та полягає у наступному.

Нехай порядок еліптичної кривої дорівнює N , e – ціле число, взаємно просте з N.

За алгоритмом Евкліда можна знайти [11]:

$$d \equiv e^{-1} \bmod N \tag{1.4}$$

За визначенням,

$$e \times d = j \times N + 1. \tag{1.5}$$

Значить, для будь-якої точки P еліптичної кривої порядку N, виконується:

$$(e \times d) \times Q = (j \times N + 1) \times P = (j \times N) \times P + P = P + O = P. \tag{1.6}$$

Тепер, використовуючи e і d і будь-яку точку P еліптичної кривої, можна обчислити:

$$Q = e \times P,$$

$$P = d \times Q,$$

де P=R.

Обчислення точки P по $e \times P$ еквівалентно вирішенню завдання дискретного логарифма для еліптичної кривої.

Розглянемо загальну схему роботи системи шифрування з відкритим ключем. Для ілюстрації даної системи розглянемо наступну ситуацію:

Необхідно забезпечити конфіденційну та захищену передачу деяких секретних даних між відправником та отримувачем використовуючи незахищений канал зв’язку,

тобто такий, в якому, з ненульовою ймовірністю, існує сторона, що здатна перехоплювати всі повідомлення, що передаються даною мережею.

Позначимо сторону відправника як *A*, сторону отримувача як *B*, повідомлення, що передається як *m*. Для організації безпечного обміну повідомленнями між *A* та *B* використовується наступна методика [12]:

1. Сторони *A* та *B* генерують пари ключів (Pub_A, $Priv_A$) для сторони *A* та (Pub_B, $Priv_B$) для сторони *B*, де (*Pub, Priv*) – публічний та приватний ключі.

2. Сторона *B* надсилає незахищеним каналом стороні *A* свій публічний ключ – Pub_B.

3. Сторона *A* виконує шифрування повідомлення m за допомогою деякої функції Enc, що приймає аргументи *Pub* та *m*. Результатом виконання даної функції є деяке значення e = *Enc(PubB,m).*

4. Сторона *A* надсилає стороні *B* незахищеним каналом зашифроване повідомлення *e.*

5. Сторона *B* отримує зашифроване повідомлення *e* , та за допомогою деякої функції *Dec* , що приймає аргументи *Priv* та *e* , отримує вихідне повідомлення m = *Dec(Pr ivB, e).*

Для зворотного зв'язку виконуються аналогічні дії, де відправником є сторона *B,* отримувачем – сторона *A*, і замість (Pub_B, $Priv_B$) виконуються маніпуляції з парою ключів *(PubA, PrivA).*

1.2.3 Опис алгоритму електронно-цифрового підпису

Ще однією сферою, де активно застосовується еліптична криптографія, є алгоритм електронно-цифрового підпису [5].

Розглянемо детальніше даний алгоритм, проілюструвавши його наступним прикладом. Нехай існує деякий відправник повідомлення, оригінальність та незмінність якого після передачі незахищеним каналом повинна гарантуватись, та отримувач, що повинен мати можливість переконатись у оригінальності та незмінності отриманого повідомлення. Позначимо відправника як A, отримувача як B, повідомлення як m.

Для виконання алгоритму електронно-цифрового підпису необхідно виконати наступні дії [13]:

1. Сторона A обирає деяке значення PrivA , що є числом великої розрядності. Обчислюється точка еліптичної кривої PubA = PrivA · G, де G – генеруюча точка еліптичної кривої, тобто точка з великим порядком. Точка PubA називається електронно-цифровим підписом відправника A.

2. Обчислюється хеш-сума повідомлення використовуючи деяку функцію Hash, h = Hash(m).

3. Використовуючи електронно-цифровий підпис Pub_A, виконується підпис хеш-суми повідомлення, використовуючи деяку функцію Sign , що приймає в якості параметрів хеш та електронно-цифровий підпис, signature = Sign (h, PubA).

4. Відправнику B надсилається (m, signatre, PubA). Для перевірки електронно-цифрового підпису на стороні отримувача виконуються наступне:

1. Обчислюється хеш повідомлення h = Hash(m).

2. Використовується деяка функція Verify , що приймає параметрами хеш-суму h , підпис повідомлення signature, електронно-цифровий підпис Pub_A, та повертає булеве значення verified = Verify (h, signature, PubA).

3. Якщо значення verified = true , то повідомлення є оригінальним та не було змінене в процесі передачі незахищеним каналом.

1.3. Алгоритми генерування та перевірки цифрового підпису

Еліптична криптографія знайшла своє застосування і в алгоритмах електронно-цифрового підпису, зокрема в алгоритмі ECDSA (elliptic curve digital signature algorithm). Розглянемо схеми генерування та перевірки електронно-цифрового підпису.

1.3.1. Схема цифрового підпису Ель-Гамаля

Схема Ель-Гамаля (ElGamal) [6] – схема шифрування з відкритим ключем, заснована на складності обчислення дискретних логарифмів. Дана схема може використовуватись як для шифрування так і як алгоритм електронно-цифрового підпису. Схема була запропонована Тахером Ель-Гамаль в 1985 році. Він удосконалив систему Діффі-Хеллмана і отримав два алгоритми, які призначені для шифрування і для аутентифікації.

При використанні в режимі електронно-цифрового підпису, одержувач підписаного повідомлення може використовувати цифровий підпис для перевірки незмінності та оригінальності повідомлення підписаного відправником. Для перевірки електронно-цифрового підпису необхідно використовувати деяку надійну хеш-функцію.

Розглянемо узагальнену схему електронного підпису Ель-Гамаля [14].

Генерація ключів:

1. Генерується випадкове просте число довжиною p бітів.

2. Вибирається випадковий примітивний елемент $g \in Zp$.

3. Вибирається випадкове ціле x таке що $1 < x < p - 1$.

4. Обчислюється $y = g^x \bmod p$.

5. Відкритим ключем є трійка (p, g, y), а закритим число x.

Алгоритм підпису повідомлення *M*:

1. Обчислити хеш-повідомлення: $m = h(M)$.

2. Вибрати випадкове число $1 < k < p - 1$ взаємно просте з $p - 1$ і обчислюється $r = g^k \bmod p$.

3. Обчислити число $s \equiv (m - xr)\, k^{-1} \bmod (p - 1)$.

4. Підписом повідомлення M є пара (r, s).

Перевірка підпису:

1. Перевіряється справедливість умов: $0 < r < p$ і $0 < s < p - 1$. Якщо

хоча б одна з них не виконується, то підпис вважається невірним.

2. Обчислюється хеш повідомлення $m = h(M)$.

3. Підпис вважається вірним, якщо виконується порівняння: $y^r r^s \equiv g^m \bmod p$.

1.3.2. Алгоритм ECDSA

ECDSA (Elliptic Curve Digital Signature Algorithm) – алгоритм з відкритим ключем для створення цифрового підпису, аналогічний, за своєю будовою, DSA, але визначений, на відміну від нього, не над полем цілих чисел, а у групі точок ЕК. Розглянемо використання алгоритму ECDSA з використанням еліптичної кривої з параметрами, визначеними над полем GF (p) [15].

Генерація ключів ECDSA.

Нехай E – еліптична крива (ЕК), визначена над полем GF (p) і P – точка високого порядку q кривої E. Відправник A конструює свій ключ, виконуючи наступні кроки:

1. Обирає випадкове ціле число x з інтервалу [1, q - 1].

2. Обчислює добуток $Q = x \cdot P$.

Відкритим ключем відправника A є точка Q , а закритим – x .

Обчислення цифрового підпису

Для того, щоб підписати деяке повідомлення m відправник A повинен зробити наступне [7]:

1. Обрати випадкове ціле число $k \in [1, q - 1]$.

2. Обчислити $k \cdot P = (x_1, y_1)$ та $r = x_1 \bmod q$. Якщо $r \equiv 0 (\bmod q)$, то

обирають нове випадкове число $k \in [1, q - 1]$.

3. Обчислити k^{-1} (mod q) та s k^{-1} (h+x+r) mod q, де h – значення хеш-функції повідомлення, що підписується. Якщо s=0, то значення s^{-1}mod q не існує, тому необхідно повернутись до п.1.

Підписом для повідомлення є пара цілих чисел (r, s).

Перевірка цифрового підпису.

Для того, щоб перевірити підпис відправника A на повідомлення, отримувач B повинен зробити наступне:

1. Отримати копію відкритого ключа Q відправника A.

2. Перевірити, що числа r та s є цілими числами з інтервалу [1, q – 1].

та обчислити значення хеш-функції h від повідомлення.

3. Обчислити $u_1=s^{-1}$ h mod q та 1 $u_2 = s^{-1}$ r mod q.

4. Обчислити $u_1 \cdot P + u_2 \cdot Q = (x_0, y_0)$ та $v = x_0 \bmod q$.

5. Прийняти підпис, якщо v = r .

Можна легко показати, що даний алгоритм дійсно успішно засвідчує цифровий підпис. Якщо підпис (r, s) повідомлення m було дійсно згенеровано з використанням секретного ключа, то s k-1(h + xr) (mod q).

Розкриваючи дужки отримаємо:

$k \equiv s^{-1} \cdot (h + xr) \equiv s^{-1} \cdot h + s^{-1} rx = u_1 u_2 x \pmod q$.

Тоді $u_1 \cdot P + u_2 \cdot Q = (u_1 + u_2 x) \cdot P = kP$ і значить v = r , що і вимагається для підтвердження підпису.

1.4 Висновки до розділу 1

У першому розділі магістерської роботи проведено огляд криптосистем заснованих на еліптичних кривих, що довели свою цінність та високий рівень захищеності. Розглянуто ряд існуючих протоколів розподілу ключів в асиметричних криптосистемах, а також алгоритмів електронно-цифрового підпису на еліптичних кривих.

У методах шифрування еліптичної криптографії та схемах цифрового підпису, що ґрунтуються на властивостях адитивної абелевої групи, утвореної точками еліптичної кривої, так само, найбільш вживаною є операція множення точки еліптичної кривої на число.

Зазначено, що використовуючи еліптичні криві, необхідно виконати такі арифметичні операції над її точками, як додавання, подвоєння та множення точок еліптичної кривої. Зокрема, в алгоритмі ECDSA, під час виконання алгоритму перевірки електронно-цифрового підпису необхідно виконувати обчислення виду $kP + lQ$, що і

називається проблемою багатократного скалярного множення [8] точок еліптичної кривої на число.

Під час досліджень, проведених в рамках даної роботи, доведено, що операція багатократного множення точок еліптичної кривої на число, є найбільш часо- та ресурсозатратною. Таким чином, підходи, що дозволяють прискорити дане обчислення, представляють велику практичну цінність.

РОЗДІЛ 2
АНАЛІЗ ОПЕРАЦІЙ НАД ТОЧКАМИ ЕЛІПТИЧНИХ КРИВИХ

В межах криптографії, активне застосування знаходять еліптичні криві з параметрами, визначеними над скінченними полями, а саме полями з простою характеристикою – GF(p) та бінарним розширенням – $GF(2^m)$. Це в свою чергу означає, що координати точок еліптичної кривої належать деякому скінченному полю, що знімає проблему округлення значень при виконанні операцій над ними [16].

В даному розділі розглядаються математичні основи теорії скінченних полів або полів Галуа, зокрема, полів GF(p) та $GF(2^m)$, арифметичних дій над її елементами, арифметичні операції над точками еліптичної кривої та особливості програмної реалізації зазначених компонентів. Вводяться поняття характеристики поля, мультиплікативної групи, незвідного полінома, додавання, віднімання, множення елементів поля, знаходження оберненого елемента тощо. Для подання однозначного і вичерпного визначення скінченного поля необхідно дати декілька математичних визначень, що використовуються для формулювання означення поля.

2.1. Алгебраїчні операції в скінченних полях

Група – це алгебраїчна структура, у якій визначена операція множення елементів даної множини та результат якої належить тій же структурі [9]. Перетворення в групі задовольняють такі властивості як:

Асоціативність: для довільних елементів a, b, c групи G виконується правило (ab)c = a(bc).

Існування нейтрального елемента: існує елемент e такий, що для кожного елемента a групи G виконується ae = ea = a.

Існування оберненого елемента: для кожного елемента a групи G існує елемент a^{-1} такий, що $aa^{-1} = e$.

Кільце – це алгебраїчна структура, в якій визначені операції додавання та множення з властивостями, подібними до додавання і множення цілих чисел. По суті кільце - це група з додатковою операцією додавання [10]. Перетворення в кільці в додачу до групи задовольняють такі властивості як:

Дистрибутивність: для довільних елементів a , b , c групи G виконується правило (a + b)c = ac + bc.

Існування нейтрального елемента: існує елемент e такий, що для кожного елемента a групи G виконується ae = ea = a.

Існування адитивної групи кільця і нейтральний елемент в ній позначають як 0.

Скінченне поле або поле Галуа поле, яке складається зі скінченної множини елементів. Ненульові елементи поля утворюють групу щодо операції множення, яка називається мультиплікативною групою поля. Ця група є циклічною, тобто вона має твірний елемент, а всі інші елементи отримуються шляхом піднесення до степеня твірного. Найменше поле Галуа містить лише два елементи – 0 та 1, арифметичні операції над якими відбуваються за модулем 2 [11].

Для будь-якого простого числа p кільце залишків mod(p) – це скінчене поле з елементів, яке позначається GF(p).

Елементи цього поля можуть бути представлені цілими числами, для них визначені операції додавання та множення за модулем p . Будь-яке скінчене поле містить p^n елементів і однозначно задається своєю характеристикою p і ступенем n . Загалом, все описане вище, можна сформулювати наступним чином. Полем F (+,*) є множина F , на якій визначені дві операції: додавання та множення, для яких виконуються наступні аксіоми:

1. Комутативність додавання $\forall a,b \in F : a + b = b + a$.
2. Асоціативність додавання $\forall a,b,c \in F : (a + b) + c = (b + c) + a$.
3. Існування нульового елемента $\exists 0 \in F : \forall a \in F \; a + 0 = a$.
4. Існування оберненого елемента відносно додавання $\forall a \in F \; \exists - a \in F : a + -a = 0$.
5. Комутативність множення $\forall a,b \in F : a*b = b*a$.
6. Асоціативність множення $\forall a,b,c \in F : (a*b)*c = (b*c)*a$.
7. Існування нейтрального елемента $\exists e \in F : \forall a \in F \; a*e = a$.
8. Існування нейтрального елемента відносно множення $\forall a \in F \; \exists a^{-1} \in F : a * a^{-1} = e$.
9. Дистрибутивність додавання відносно множення $\forall a,b,c \in F : (a + b)*c = a*c + b*c$.

Побудову скінченного поля можна проілюструвати конкретним прикладом.

Нехай, необхідно побудувати скінченне поле з 16 елементів.

Отримаємо $GF(16) = GF(2^4)$, для побудови елементів поля в такому випадку необхідно обрати незвідний багаточлен ступеня 4, тобто такий, що не розкладається на множники. Таким многочленом є вираз $x^4 + x + 1$.

Тоді, елементи поля задаються у вигляді многочленів – залишок від ділення елемента, піднесеного до ступеня на незвідний многочлен. Побудову елементів поля $GF(2^4)$ проілюстровано в табл. 2.1.

Таблиця 2.1 – Елементи поля $GF(2^4)$

Багаточлен	Ступінь	$1, x, x^2, x^3$
α	α	(0,1,0,0)
α^2	α^2	(0,0,1,0)
α^3	α^3	(0,0,0,1)
$1+\alpha$	α^4	(1,1,0,0)
$\alpha+\alpha^2$	α^5	(0,1,1,0)
$\alpha^2+\alpha^3$	α^6	(0,0,1,1)
$\alpha^3+\alpha+1$	α^7	(1,1,0,1)
α^2+1	α^8	(1,0,1,0)
$\alpha^3+\alpha$	α^9	(0,1,0,1)
$\alpha^2+\alpha+1$	α^{10}	(1,1,1,0)
$\alpha^3+\alpha^2+\alpha$	α^{11}	(0,1,1,1)
$\alpha^3+\alpha^2+\alpha+1$	α^{12}	(1,1,1,1)
$\alpha^3+\alpha^2+1$	α^{13}	(1,0,1,1)
α^3+1	α^{14}	(1,0,0,1)
1	α^{15}	(1,0,0,0)

Для знаходження мультиплікативно оберненого елемента (позначимо як *inv*), необхідно застосувати розширений алгоритм Евкліда.

Нехай a та b цілі числа, причому $0 < b \le a$.

Тоді класичний розширений алгоритм Евкліда полягає у виконанні наступних кроків:

1. $u := a; v := b ; A:=1; B:= 0 ; C := 0 ; D:=1.$

2. Поки $v \neq 0$

$q:=[u/v]$
$t_1 := u - q \cdot v$; // *u mod v*

$t_2 := A - q \cdot C$;

$t_3 := B - q \cdot D$;

$u := v$;

$A := C ; B := D. v := t_1 ; C := t_2 ; D := t_3$

3. $d := u, x := A; y := B$.

Результат: d , x , y , inv .

2.2 Додавання та подвоєння точок еліптичної кривої

Введемо наступні правила додавання точок еліптичної кривої [17]:

1. Точка O виступає в ролі нульового елементу. Таким чином, $O = - O$ та для будь-якої точки P на еліптичній кривій $P + O = P$.

2. Вертикальні лінія перетинає криву в двох точках з однією й тією ж координатою x . Наприклад, $S = (x, y)$ та $T = (x, -y)$. Ця пряма перетинає криву і в нескінченно віддаленій точці. Тому $P_1 + P_2 + O = O$ та $P_1 = - P_2$.

3. Для того, щоб додати дві точки P та Q з різними координатами x , необхідно провести через ці точки пряму та знайти точку перетину її з ЕК. Якщо пряма не є дотичною до кривої в точках P або Q , то існує тільки одна така точка, позначимо її S . Відповідно до нашого припущення, $P + Q + S = 0$, $P + Q = - S$ або $P + Q = T$. Якщо пряма є дотичною до кривої в якій-небудь з точок P або Q , то в цьому випадку варто покласти $S = P$ або $S = Q$ відповідно.

4. Щоб подвоїти точку Q, варто провести дотичну в точці Q та знайти іншу точку перетину S з еліптичною кривою. Тоді $Q + Q = 2Q = - S$.

Введена таким чином операція додавання підпорядковується всім звичайним правилам додавання: комутативному та асоціативному законам.

Множина точок ЕК має такі властивості [18]:

1. $P + O = P$;

2. Якщо $P = (x, y)$, то $P + (x, -y) = O$. Точка $(x, -y)$ є від'ємним значенням точки P і позначається $-P$. Зазначимо, що $(x, -y)$ також лежить на еліптичній кривій. Якщо $P = (x_1, y_1)$ та $Q = (x_2, y_2)$, де $P \neq Q$, то $P + Q = (x_3, y_3)$ визначається за наступними формулами:

$$x_3 \equiv \lambda^2 - x_1 - x_2 \pmod{p}; \tag{2.1}$$

$$y_3 \equiv \lambda(x_1 - x_3) - y_1 \pmod{p}, \tag{2.2}]$$

$$\text{де } \lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & \text{якщо } P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1}, & \text{якщо } P = Q. \end{cases}$$

Число λ є кутовим коефіцієнтом січної [13], проведеної через точки P та Q

При P = Q січна перетворюється в дотичну, чим і пояснюється наявність двох формул для обчислення λ.

У випадку визначення параметрів еліптичної кривої над деяким скінченним полем, всі арифметичні операції над даними параметрами підпорядковуються законам даного скінченного поля.

Наприклад, у випадку $GF(p)$, всі обчислення проводитимуться за модулем p , а у випадку $GF(2^m)$ – за модулем незвідного многочлена для даного скінченного поля [19].

2.3 Особливості програмної реалізації операцій над точками еліптичної кривої

Еліптичні криві з параметрами, що визначені над скінченними полями мають ряд істотних переваг у порівнянні з тими, що визначені над раціональними числами, а саме – теоретична ефективність реалізації в обчислювальній техніці. Це обумовлено тим, що у випадку поля $GF(p)$, числа можуть бути легко поданими у вигляді бітових слів, адже обчислювальні пристрої оперують саме з числами у двійковому поданні, а операція обчислення за модулем може бути представлена як звичайна булева функція XOR. Для поля $GF(2^m)$, многочлени, що є елементами поля можуть бути представлені у вигляді кодових двійкових слів, де кожен розряд позначає коефіцієнт при доданку-ступені.

Наприклад, многочлен $\alpha + \alpha + 1$ може бути представлений на апаратному рівні у вигляді кодового слова «1011» або $1*\alpha + 1*\alpha + 0*\alpha + 1*\alpha = \alpha + \alpha + 1$ [20].

Дане представлення дозволяє оперувати елементами поля як звичайними цілочисельними значеннями, коли додавання, віднімання і т.д. виконуються за модулем 2, тобто використовуючи ту саму булеву логічну функцію XOR. Що стосується алгоритмів, що необхідні для знаходження елементів поля та виконання арифметичних операцій над ними, як наприклад, алгоритм Евкліда, то даний алгоритм є за своєю природою ітераційним, що означає можливість його ефективної програмної реалізації для електронно-обчислювальних пристроїв.

Що стосується програмної реалізації операцій над точками еліптичної кривої, то слід зазначити, що операції над точками еліптичної кривої є за своєю суттю комбінацією арифметичних операцій над параметрами еліптичної кривої та координатами її точок. Таким чином еліптична крива в обчислювальній техніці може бути однозначно представлена парою цілочисельних значень координат (x; y). Окремо слід зазначити випадок операцій за участю так званої точки на нескінченності.

Складність даного випадку полягає у тому, що точка на нескінченності є скоріше математичною концепцією, адже у цієї точки невизначені координати (нескінченність), що очевидно не належать скінченному полю, над яким визначена дана крива. Таким чином, результати операцій за участю точки на нескінченності повинні бути оброблені як граничні випадки, тобто аксіоматично визначеними в тілі алгоритму, а саме [21]:

1. $O = -O$
2. $O + P = P$
3. $k \cdot O = O$

З точки зору об’єктного проектування, операції над точками еліптичної кривої є за своєю суттю надбудовою над операціями в скінченних полях. Деталі програмної реалізації даних модулів будуть детальніше описані в розділі 4.

2.4 Множення точок еліптичної кривої на 3 та 4

Розглянемо докладніше алгоритми множення точки еліптичної кривої на 3 та 4 для еліптичної кривої у формі Вейерштраса [22].

Найпростішим способом виконання обчислення $3 \cdot P$ та $4 \cdot P$ [14], є подвійне подвоєння та додавання плюс подвоєння. В той же час існує інший підхід, який

ґрунтується на обчисленні $3 \cdot P$ та $4 \cdot P$, використовуючи тільки координати точки P. Існують три найкращі способи обчислення $3 \cdot P$ та $4 \cdot P$ [23].

Спосіб 1:

$t_1 = 2y_1^2;$

$t_2 = 3x_1^2 + a;$

$t_3 = t_2^2;$

$d = t_1(3 \cdot x_1) - t_3;$

$t_4 = d(2y_1);$

$inv = t_4^{-1};$

$\lambda_1 = d \cdot inv \cdot t_2;$

$\lambda_2 = t_1 \cdot inv - \lambda_1;$

$y_3 = (x_1 - x_3)\lambda_2 - y_1.$

Спосіб 2:

$inv_1 = (2y_1)^{-1};$

$\lambda_1 = (3x_1^2 + a) \cdot inv_1;$

$inv_2 = ((3x_1^2 + a) - 12x_1y_1)^{-1};$

$\lambda_2 = -\lambda_1 - 8y_1^3 \cdot inv_2;$

$x_3 = (\lambda_2 - \lambda_1) \cdot (\lambda_2 + \lambda_1) - y_1;$

$y_3 = -\lambda_2 (\lambda_2 - \lambda_1) \cdot (\lambda_2 + \lambda_1) - y_1;$

На рисунку 2.1 наведено приклад додавання точок на еліптичній кривій Вейєрштрасса

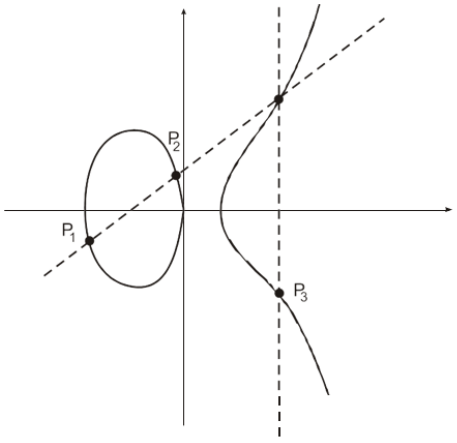


Рисунок 2.1 – Операція додавання точок на еліптичній кривій Вейєрштрасса

У даних алгоритмах, при обчисленні мультиплікативно оберненого елемента необхідно перевіряти аргумент на рівність нулю. Нехай P = (x1, y1) , і необхідно обчислити $4 \cdot P = (x4, y4)$ [24].

Припустимо, що: $d = (3x^2 + a)(12x_1y_1 - 3(x^2 + a^2) - 8y_1^4$.

Легко побачити, що $d = 8y_1y_3$, де y_3 – це координата у результуючої точки. Обчислення значення d потребує одну операцію множення та 5 операцій піднесення до квадрату. Якщо a та b є малими, тоді d може бути обчислено за 4 операції піднесення до квадрату (вважаємо, що a^2, a^3 та $27b^2$ обчислюються попередньо та зберігаються як [25].

$$d = y_1^4 + 3ax_1^4 - 6a^2x_1^2 + 18by_1^2 - 24abx_1 - a^3 - 27b^2 . \qquad (2.3)$$

Визначивши $D = (2y_1)d$ та $I = D^{-1}$, отримаємо:

$$\frac{1}{2y_1} = dI, \quad \frac{1}{2y_3} = 8y_1^4 I.$$

Таким чином, спосіб 3 полягає у наступному:

$t_1 = x_1^2$;

$t_2 = 3x_1^2 + a$;

$t_3 = 2y_1^2$;

$t_4 = t_3^2$;

$t_5 = (x_1 + t_3)^2 - t_1 - t_4$;

$d = t_2(3t_5 - t_2^2) - 2t_4$;

$t_6 = d(2y_1)$;

$inv = t_6^{-1}$;

$\lambda_1 = d \cdot inv \cdot t_2$;

$x_3 = \lambda_1^2 - 2x_1$;

$t_7 = 3x_3^2 + a$;

$\lambda_2 = 2t_4 \cdot inv\, t_7$;

$x_4 = \lambda_2^2 - 2x_3$;

$y_4 = \lambda_2(x_3 - x_4) - y_3$.

Скінченні поля представляють особливий інтерес з огляду на ефективність їх застосування в апаратних та програмних реалізаціях криптосистем заснованих на еліптичній кривій, через свою очевидну близькість до апаратного (двійкового) подання

значень та виконання операцій на обчислювальних приладах. Проаналізовані скінченні поля такі як $GF(p)$ та $GF(2^m)$, принципи їх побудови та операції над їх елементами .

Продемонстровано спосіб знаходження оберненого елемента в мультиплікативній групі поля за допомогою розширеного алгоритму Евкліда. Продемонстровано взаємозв'язок між теорією скінченних полів та операцій над точками, описаний особливий випадок операцій з точкою на нескінченності тощо.

Таким чином, можна зробити висновок, що оптимізація операцій з точками еліптичної кривої з параметрами, визначеними над скінченними полями $GF(p)$ та $GF(2^m)$, мають особливий практичний інтерес, адже в реаліях сучасного рівня розвитку обчислювальної техніки криптографічні схеми майже завжди розглядаються з огляду на їх потенційне використання у сфері інформаційних технологій.

Отже, скінченне поле $GF(2^m)$ якнайкраще підходить до використання в обчислювальній техніці, оскільки наслідує принципи виконання операцій в ПК.

2.5 Модифікація методу багатократного скалярного множення точок еліптичних кривих

Методи багатократного скалярного множення точок еліптичних кривих на число мають особливу важливість в контексті застосування алгоритму електронно-цифрового підпису з використанням еліптичних кривих. Окрім стандартного обчислення, розрізняють ряд підходів, що дозволяють значно пришвидшити даний процес. Умовно, дані методи можна розділити на 4 основні групи: методи, що ґрунтуються на передобчисленні результатів багатократного множення точок; методи що ґрунтуються на оптимізації представлення множників багатократного множення точок; методи що ґрунтуються на двобазисному представленні множників багатократного множення точок; метод так званого «плаваючого вікна» [25].

В даному розділі буде детально розглянуто вищезгадані методи, буде проведений їх порівняльний аналіз та описано запропоновану модифікацію методу багатократного скалярного множення точок еліптичних кривих на число.

На сьогоднішній день, існує безліч наукових праць та публікацій, присвячених оптимізації методів багатократного скалярного множення точок еліптичної кривої на число. В даному підрозділі наведено ряд методів, що демонструють принципи, спільні до тих, що були використані при розробці запропонованої модифікації.

2.6 Метод SMPM

Метод SMPM – одночасне множинне множення точок (simultaneous multiple point multiplication).

В даному методі числа k та l представляються у вигляді блоків по w біт, кількістю $d = \lceil t/w \rceil$, виконується передобчислення точок w $iP + jQ, 0 \le i, j < 2$. Тоді, числа k та l представляються у вигляді $k = K_{d-1} K_{d-2} ... K_1 K_0$, $L = L_{d-1} L_{d-1} ... L_1 L_0$.

Для обчислення бажаного результату необхідно обчислити $R = 2w R + (Ki P + LiQ)$ $d - 1$ раз [15].

Алгоритм виконання [26]:

1. Обчислити $iP + jQ$ для всіх $I, j \in [0, 2^{1-w}]$,.

2. Записати $k = K_{d-1} K_{d-2} ... K_1 K_0$, $L_{d-1} L_{d-1} ... L_1 L_0$, де всі Ki та Li мають довжину w біт та $d = \lceil t/w \rceil$.

3. $R \leftarrow \infty$.

4. Для кожного i від $d - 1$ до 0 виконати $R \leftarrow 2^{\omega} R$.

5. Повернути R.

Даний метод має істотний недолік [16] – швидкодія даного методу обернено пропорційна кількості передобчислень. Це означає, що для досягнення великої швидкодії необхідно виконати нераціонально велику кількість передобчислень, для зберігання яких, необхідно виділити постійну пам’ять у нераціонально великих розмірах. Дана закономірність зображена на рисунку 2.2 [27]

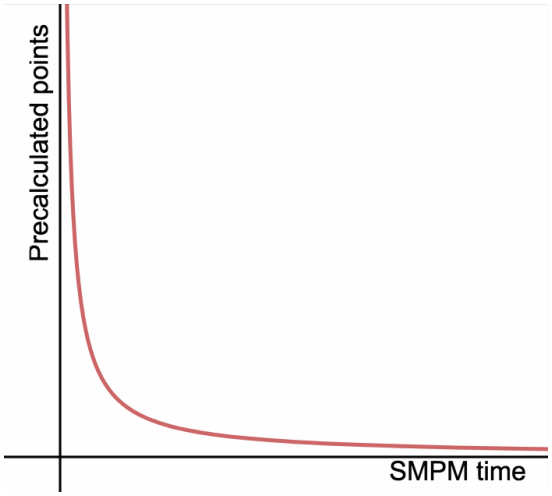


Рисунок 2.2 – Залежність швидкодії методу від кількості передобчислених точок

2.7 Метод SMPM з використанням NAF та JSF

З використанням фіксованого розміру блоків, в середньому виконується $3t / 4$ операцій додавання точок для обчислення $kP + lQ$. При використанні подання NAF, кількість операцій додавання знижується до $5t / 9$. При використанні ж форми JSF, дана кількість знижується до $t / 2$. При використанні даних підходів необхідно виконати передобчислення точок ($\pm P \ \pm Q$). При використанні JSF кількість операцій подвоєнь дорівнює t [28].

Нижче приведений алгоритм знаходження форми JSF для довільних k_1та k_2.

1. $l \leftarrow 0$, 0 1 $d \leftarrow$, 0 2 $d \leftarrow$

2. Доки ($k_1 + d_1 > 0$ або $k_2 + d_2 > 0$)

2.1. $l_1 \leftarrow d_1 + k_1$, $l_2 \leftarrow d_2 + k_2$

2.2. Для i від 1 до 2 виконати:

2.2.1. Якщо $li \bmod 2 \equiv 0$ $u = 0$ інакше

2.2.1.1 mod 4 i $u \leftarrow l$

2.2.1.2 Якщо mod8 $\equiv \pm 3$ i l та mod 4 2 3 $\equiv -i$ l тоді $u \leftarrow -u$

2.2.2. $u \leftarrow$ l i *mod 4*

2.3. Для i від 1 до 2 виконати:

2.3.1. Якщо I l $2d = 1 + k$ i тоді $d \leftarrow 1 - d$ i

2.3.2. k $i = k$ i /2

2.4. $l \leftarrow l + 1$

3. Повернути $JSF(k_1, k_2)$.

При використанні даного підходу кількість передобчислень зменшується до чотирьох: $\{ P + Q; P - Q; -P - Q; -P + Q\}$. Перевагою JSF над NAF є визначеність розподілу ненульових елементів подання для визначених k та l. Наприклад, числа 53 та 102 в поданні NAF мають об’єднану відстань Хемінга $H = 8$, в той час як для подання JSF об’єднана відстань Хемінга становить $H = 6$.

При обчисленні $R = kP + lQ$ з використанням JSF та NAF використовується форма $k_i = K_{d-1} K_{d-2} ... K_1 K_0$, $L_{d-1} L_{d-1} ... L_1 L_0$ – де k_i та $L_i \in \{-1, 0, 1\}$. Обчислюючи $R \leftarrow 2R + (k_i P + l_i Q)$ отримуємо результат [17].

2.8 Метод, що базується на використанні форми DBNS

Двобазисна система подання числа (англ. DBNS- double based number system) - це схема подання чисел, при якій кожне невід'ємне ціле число n представляється у вигляді сум та різниць добутків степенів 2 та 3 [29].

Математично дане представлення можна зобразити як:

$$n = \sum_{i=1}^{M} s_i 2^{b_i} 3^{t_i} \tag{2.4}$$

при $s \in \{1,-1\}$.

Для довільного цілого числа, в загальному випадку, існує велика кількість варіантів подання у формі DBNS. Наприклад, лише для $s=1$число 100 має рівно 427 різних варіантів форми DBNS, а число 1000 – рівно 1,295,579 DBNS-представлень. Серед всіх існуючих варіантів подання DBNS існує таке, що має мінімальну кількість доданків. Таке представлення називається канонічним [30].

В описаному в даному розділі методі необхідно обчислювати кількість елементів DBNS-подання, що менші за деяке число 2^n. Дане значення дозволяє знайти необхідну кількість подвоєнь та потроєнь точок, необхідних для обчислення кінцевого результату.

В даному методі, так само як і в методі, описаному в пункті 3.1.1, числа k та l подаються у вигляді блоків довжиною ω біт. Виконується передобчислення всіх елементів подання DBNS, що менші за 2^{ω} для точок Q , $P+Q$ та $P-Q$. Таким чином, замість операції множення точки на число, з'являється можливість виконати додавання множників виду [31]

$$kP + lQ = \sum_{i=0}^{M} s^1{}_i 2^{k1_i} 3^{t_i^1} P + s^2{}_i 2^{k^2{}_i} 3^{t_i^2} Q. \tag{2.5}$$

Графічно процес обчислення в даному методі представлений на рисунку *2*.3.

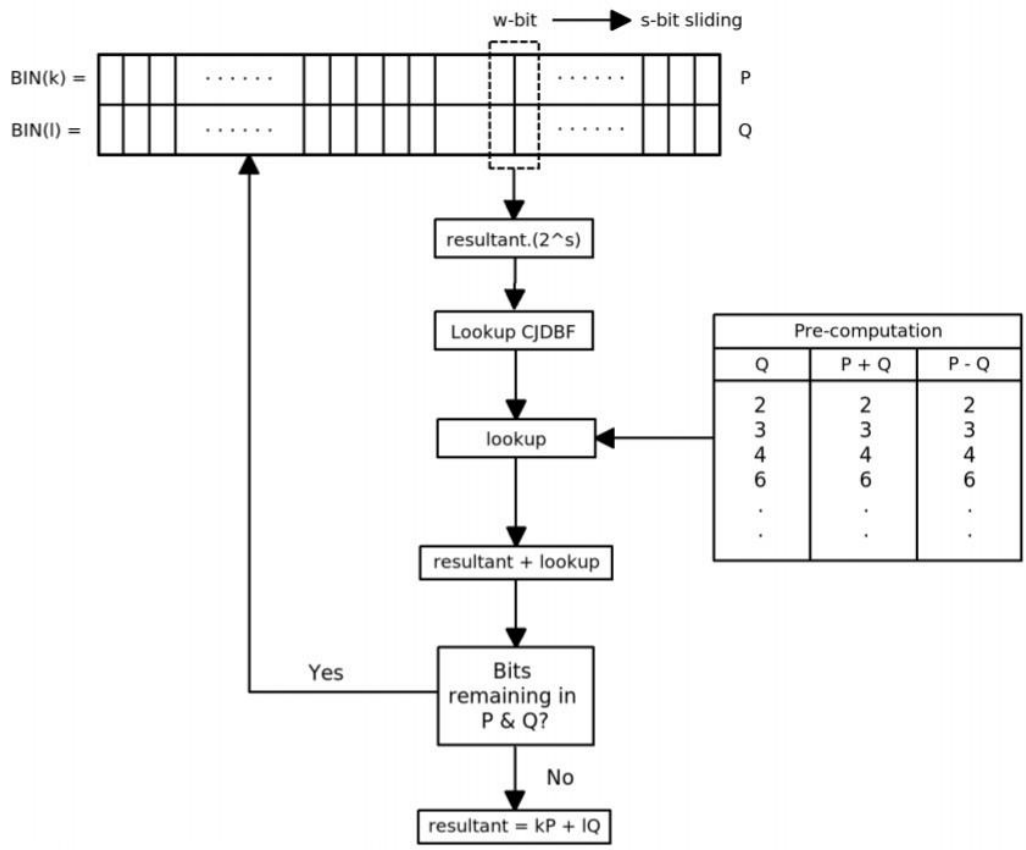


Рисунок 2.3 – Схема роботи методу, що базується на використанні форми DBNS

2.9 Модифікований метод на основі подання чисел у вигляді JSF

В методі, описаному в пункті 3.1.2. пропонується передобчислення точок еліптичної кривої: $\{P + Q; P - Q; -P - Q; -P + Q; P; Q; -P; -Q\}$ з виконанням зсуву вліво на один розряд результату після кожної ітерації [20].

Пропонується виконання передобчислення зсувів для всіх комбінацій значень $d_i P + d_j Q,\ d\,i\,j \in \{-1, 0, 1\}$.

Таким чином, виконується передобчислення 2^n ($d_i P + d_j Q,\ d\,i\,j \in \{-1, 0, 1\}$), $\in \{-1, 0,1\}$.

Наприклад, для $n = 2$, таблиця передобчислень містить 2*8 = 16 значень.

Для обчислення значення $R=53P+102Q$ виконуються наступні дії:

1. Отримати подання JSF для чисел $k = 53$ та $l = 102$. $k = \{1, 0, 0, -1, 0, -1, -1\}$, $l = \{1, 0, -1, 0, 0, 1, 1, 0\}$.

2. Для кожного номера розряду $i=2$ отримати значення з таблиці передобчислених точок, тобто значення $Val=2^2\ 0\ P + 2^2(-1)\ Q = -4\ Q$.

Додати до результату. Наприклад, для $i = 2$, $R = R + Val$, де

При використанні даної модифікації, час перевірки електронно-цифрового підпису значно зменшується для еліптичних кривих з параметрами, визначеними над полем $GF(2^m)$. Це пов'язано з тим, що при використанні даної модифікації, всі операції множення відбуваються на етапі формування таблиці передобчислень, а в процесі обчислення результату, виконується лише операція додавання точок еліптичної кривої. Назвемо запропоновану модифікацію методу багатократного множення точок еліптичної кривої JSFImp (JSF Improved).

2.10 Порівняльний аналіз модифікованого та відомих методів

Для виконання порівняння методів багатократного множення точки еліптичної кривої на число проведено замір часу перевірки електронно-цифрового підпису за алгоритмом ECDSA із застосуванням існуючих методів багатократного множення точок еліптичної кривої та запропонованої модифікації. В якості вхідних даних використані всі еліптичні криві, що запропоновані стандартом SEC 2 (Recommended Elliptic Curve Domain Parameters) [32] (табл. 2.2, табл. 2.3, рис. 2.4).

Таблиця 2.2 – Значення часу перевірки електронно-цифрового підпису для еліптичних кривих з параметрами визначеними над полем $GF(p)$, (мс)

	SMPM4	SMPM5	SMPM6	SMPM7	NONE	JSF	NAF	JSFImp
SECP112R1	11,9	10,6	7,2	7,9	9,2	16,7	12,8	4,2
SECP112R2	4,3	3,9	6,7	5,1	7,3	16,4	16,3	3,2
SECP128R1	11,1	5,5	6,2	8,5	7,9	10,4	25,3	5
SECP128R2	5,3	5,2	4,8	25,4	13,4	9,5	11,6	4,3
SECP192K1	19,3	13,9	18,3	12,5	14,7	26,4	29,4	11,4
SECP192R1	13,8	14,6	17,1	20,5	16	22,5	22,1	11,6
SECP224K1	31,3	16,4	15,5	13,8	19,8	30,9	28,8	16,6
SECP224R1	21,2	13,9	14	14,8	25,3	30,1	32,3	17,9
SECP256K1	19,2	25,8	18,5	19,4	25,8	42,8	46	25
SECP256R1	18,8	18,4	19,5	23,3	29,7	54,6	40,7	23,5
SECP384R1	49,9	50,8	46,4	47	68,3	108,2	114,9	66,8
SECP521R1	110,6	123,1	107,9	106,1	145	246,3	238,7	166

Таблиця 2.3 – Значення часу перевірки електронно-цифрового підпису для еліптичних кривих з параметрами визначеними над полем GF(2^m), (мс)

	SMPM4	SMPM5	SMPM6	SMPM7	NONE	JSF	NAF	JSFImp
SECT113R1	119,63	132,76	113,79	107,37	394,04	272,85	212,08	26,72
SECT163K1	382,31	448,27	343,17	308,83	417,16	437,67	398,19	71,92
SECT163R1	348,89	284,97	329,49	310,63	385,1	378,53	349,87	82,76
SECT163R2	349,55	284,17	296,77	336,07	399,61	333,97	349,3	77,83
SECT233K1	882,59	1027,2	683,72	1066,35	922,84	829,23	799,91	164,93
SECT233R1	1066,4	1052,14	969,35	818,54	1241,4	901,1	880,04	172,2
SECT239K1	791,88	807,24	834,56	751,31	1255,24	1061,03	1089,73	191,7
SECT283K1	1141,25	1133,76	1166,88	1132,89	1852,08	1326,8	1339,85	302,2
SECT283R1	1179,77	1123,88	1136,88	1216,29	1558,77	1470,38	1479,51	296,15
SECT409K1	3187,36	3244,52	3235,17	3044,42	4202,8	3540,57	4149,11	797,64
SECT409R1	3022,08	3307,61	2810,1	3047,54	3900,77	3670,78	3452,5	856,4
SECT571K1	7165,59	8101,03	7443,83	6889,09	10513,73	8386,25	8420,83	1811,41

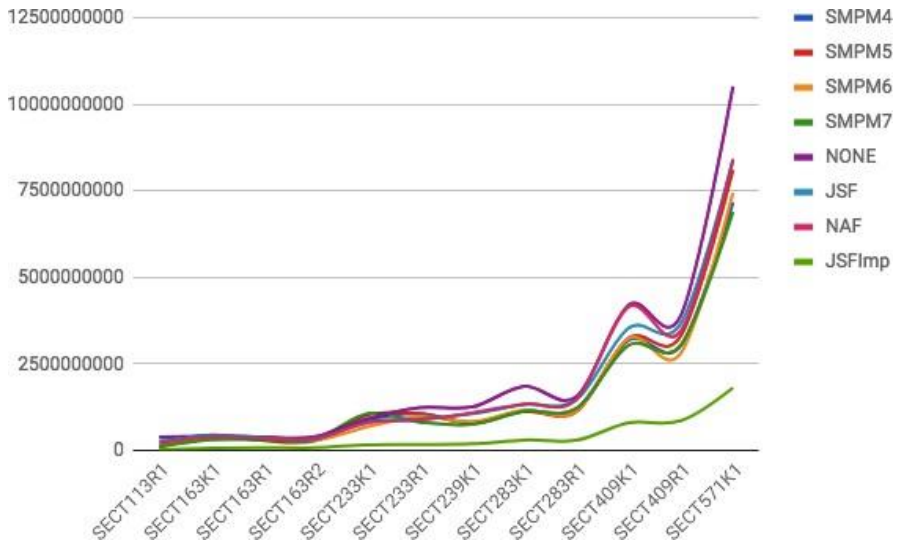


Рисунок 2.4 – Залежність часу перевірки електронно-цифрового підпису при застосуванні різних методів багатократного множення точок еліптичної кривої

2.11 Висновки до розділу 2

Запропонована модифікація алгоритму ECDSA дає вищу швидкодію при перевірці електронно-цифрового підпису за алгоритмом над еліптичною кривою з параметрами, визначеними над скінченним полем $GF(2^m)$. Обчислювальна складність виконання передобчислень для запропонованої модифікації є набагато меншою ніж при використанні методу SMPM, який показує найкращі результати серед існуючих методів.

РОЗДІЛ 3
ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ РЕАЛІЗАЦІЇ ОПЕРАЦІЙ НАД ТОЧКАМИ ЕЛІПТИЧНИХ КРИВИХ

3.1 Ієрархічні алгоритми при реалізації над точками еліптичних кривих

Основною операцією в групі точок еліптичної кривої є скалярний добуток точки на число. Як показано на рисунку 3.1, множення точки на число відбувається за допомогою операцій додавання точок і подвоєння точки [33].

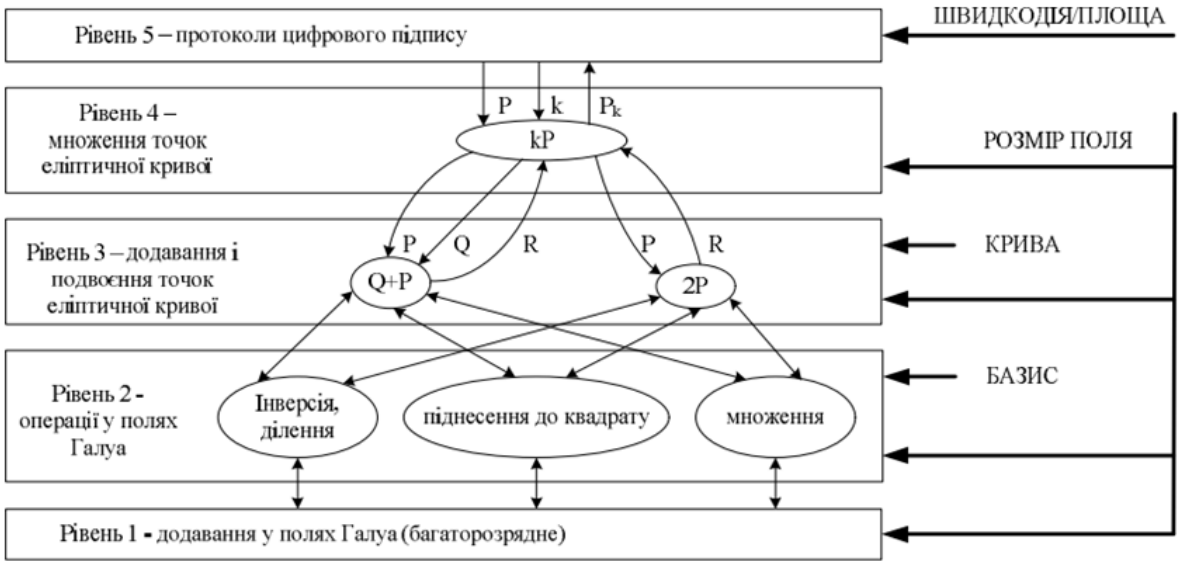


Рисунок 3.1 – Ієрархічні рівні алгоритмів

3.2 Вибір еліптичної кривої

В еліптичної криптографії використовуються т. зв. «Криві над кінцевим полем». Це означає, що криві мають кінцеве число точок. Кількість точок подібної кривої називається порядком кривої. Щоб використовувати еліптичну криву в криптографії необхідно знати її порядок. Це обумовлюється хоча б тим, що від порядку кривої залежить криптостійкість системи.

Саме в цьому і полягає вся складність. Процес вибору кривої можна записати в такий спосіб:

1. Вибір параметрів a і b, що описують рівняння

$$y^2 = x^3+ax+b \qquad (3.1)$$

2. Підрахунок точок обраної кривої.

3. Перевірка чи відповідає обрана крива із заданою кількістю точок ряду умов.

Проблема полягає в тому, що обчислення порядку еліптичної кривої є вельми нетривіальним завданням.

Найбільш поширений метод обчислення кількості точок – алгоритм Шуфа [34], що має досить велику обчислювальну складність. До того ж, алгоритм використовує дуже серйозні математичні методи і дуже складний для розуміння.

Є ще один спосіб, т.зв. метод комплексного множення – подання чисел сумою двох квадратів і еліптичні криві. Цей метод дозволяє набагато більш ефективно знаходити криві із заданою кількістю точок. Однак, на відміну від алгоритму Шуфа, який є універсальним, метод комплексного множення працює тільки при виконанні певних умов. Цей метод теж не такий простий, як може здатися спочатку [34].

NIST (Національний інститут стандартів і технології — національний орган зі стандартизації у США), з метою полегшення життя розробникам, склав список еліптичних кривих з уже відомою кількістю точок, які рекомендовано використовувати в схемах ЕЦП [35].

Для опису кривої в стандарті NIST використовується набір з 6 параметрів D = (p, a, b, G, n, h), де:

– p - просте число, модуль еліптичної кривої;

– a, b - задають рівняння еліптичної кривої;

– G - точка еліптичної кривої великого порядку. Це означає що якщо множити точку на числа менші, ніж порядок точки, кожен раз будуть виходити зовсім різні точки;

– n - порядок точки G;

– h - параметр, званий кофактор. Визначається відношенням загального числа точок на еліптичній кривій до порядку точки G. Дане число повинно бути якомога менше.

В даній роботі для прикладу використана перша рекомендована NIST крива, в якій значення описаних вище параметрів відповідно дорівнює:

p = 6277101735386680763835789423207666416083908700390324961279;

a = -3;

b = 2455155546008943817740293915197451784769108058161191238065;

Gx = 602046282375688656758213480587526111916698976636884684818

(x-координата точки G);

Gy = 174050332293622031404857552280219410364023488927386650641

(y-координата точки G);

n = 627710173538668076383578942317605901376719477318284228408 1;

h = 1.

Параметр *p*, як число, відноситься до узагальнених числах Мерсенна [ ], це означає, що його можна представити як суму різних ступенів двійки.

В нашому випадку, число p може бути записано як $p = 2^{192} – 2^{64} – 1$.

Всі еліптичні криві над полем простого числа, рекомендовані NIST можна записати таким чином. Використання подібних чисел дозволяє прискорити операцію множення за модулем великого числа.

Суть методу зводиться до подання результату множення у вигляді машинних слів довжиною 32 біта, комбінування яких дає в результаті значення по модулю великого числа.

Ще один цікавий момент пов'язаний з координатами точок. Найчастіше, в різного роду специфікаціях утворює точка G еліптичної кривої задається в стислій формі. Наприклад, в нашому випадку точку G можна задати наступним чином:

G = 0x 03188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012

Перший байт зберігає в собі дані про парність *y*-координати. Він може дорівнювати 2 (це означає що ^{y}-координата парна) або 3 (відповідно непарна). Решта байтів зберігають *x*-координату.

Маючи у своєму розпорядженні ці дані можна відновити *y*-координату наступним чином. Знаємо, що точка G належить еліптичній кривій. Відповідно для неї виконується рівність:

$$y^2 = x^3 + ax + b \bmod p \qquad (3.2)$$

Так як результатом обчислення квадратного кореня по модулю p є два числа *y* і *p-y*, обираємо те число, парність якого збігається з парністю першого байта в стислому запису координат G.

В [36] наведені часові характеристики створення та перевірки ЕЦП.

3.3 Вибір мови програмування

Для програмної реалізації запропонованих методів оптимізації багатократного скалярного множення точок еліптичної кривої на число необхідно провести дослідження програмних засобів, підходів та комплексів, які б забезпечили найбільш оптимальні можливості для успішної їх практичної реалізації.

В усіх асиметричних криптосистемах найважливішою складовою є секретний та відкритий ключі, за допомогою яких і відбувається шифрування/дешифрування, створення та перевірка цифрового підпису. Ці ключі мають бути досить великих розмірів (наприклад, по 1024 біта кожен [25]) для забезпечення криптостійкості алгоритму. Тому, при реалізації алгоритмів виконання низькорівневих та високорівневих операцій постає задача виконувати операції над дуже великими числами (більшими за розрядність класичних цілочисельних типів як int).

Іншим важливим моментом, що був врахований на стадії проектування програмного комплексу, є наявність очевидних моделей та абстракцій, що дозволяють ефективно моделювати предметну галузь. Такими абстракціями є скінченні поля, елементи скінченних полів, еліптичні криві тощо. Дані абстракції мають різну реалізацію, як наприклад скінченні поля $GF(p)$ та $GF(2^m)$, де в одному випадку елементи можуть бути подані у вигляді цілочисельних значень, а у іншому – у вигляді многочленів, мають різну логіку виконання операцій над елементами, тощо, але мають ряд спільностей, як скажімо, однакові структури викликів операцій над елементами. Так і еліптичні криві з параметрами, визначеними над різними типами скінченних полів мають за своєю суттю абсолютно однаковий шаблон для викликів операцій над їх точками [26].

Дана особливість спонукала нас до створення ефективної об’єктно- орієнтованої моделі, що дозволяла б моделювати предметну галузь з високим рівнем абстракції. Розроблена нами архітектура буде детально описана в даному розділі.

В ході досліджень виявлено три основні мови програмування, що найбільш повно забезпечують можливості ефективної реалізації означеного програмного комплексу – Java, C++ та Python.

C++ – мова програмування високого рівня з підтримкою об'єктно- орієнтованої, узагальненої та процедурної парадигм програмування. Розроблена Б'ярном Страуструпом та початково отримала назву «Сі з класами». Базується на мові C. Вперше описана стандартом ISO/IEC 14882:1998, найбільш актуальним же є стандарт ISO/IEC 14882:2014.

У 1990-х роках C++ стала однією з найуживаніших мов програмування загального призначення. Мову C++ використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм, наприклад, відеоігор. C++ суттєво вплинула на інші популярні сьогодні мови програмування: C# та Java. При створенні C++ прагнули зберегти сумісність з мовою C.

Нововведеннями C++ порівняно з C є:

– підтримка об'єктно-орієнтованого програмування через класи.
– підтримка узагальненого програмування через шаблони.
– доповнення до стандартної бібліотеки.
– додаткові типи даних.
– обробка винятків.
– простори імен.
– вбудовані функції.
– перевантаження операторів.
– перевантаження імен функцій.
– посилання і оператори управління вільно розподіленою пам'яттю.

До стандарту C++ входить також Стандартна Бібліотека Шаблонів (STL) [27]. Вона надає такі важливі інструменти, як контейнери об'єктів, ітератори що надають доступ до цих контейнерів як до масивів тощо. Крім того, STL дозволяє схожим чином працювати і з іншими типами контейнерів, наприклад, асоціативними списками, стеками, чергами.

Основним способом організації інформації в C++ є класи. На відміну від структур (struct) мови C, що складається тільки з полів, клас C++ складається з полів і функцій-членів або методів.

Основною перевагою мови програмування C++ є можливість безпосереднього виділення та управління пам'яттю. Це дозволяє створювати нові структури данних, що можуть задовольнити вимоги до такого ключового елемента нашого програмного комплексу як елемент скінченного поля.

Java – об'єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems». Програми, написані на Java компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретного типу платформи, на якій виконується програма. Під «незалежністю від архітектури» мається на увазі те, що програма, написана на мові Java, працюватиме на будь-якій підтримуваній апаратній чи системній платформі без змін у початковому коді та перекомпіляції. На противагу C++, Java з точки зору архітектури даної мови, є більш об'єктно- орієнтованою. Всі дані і дії

групуються в класи об'єктів. Виключенням з повної об'єктності є примітивні типи (int, float тощо), що дозволяють збільшити ефективність виконання алгоритмічних задач.

У Java всі об'єкти є успадкованими від головного об'єкта (Object). На відміну від C++, у Java можливе тільки одинарне успадкування, завдяки чому виключається можливість конфліктів між членами класу (методи і змінні), які успадковуються від базових класів. Таким чином, мова програмування Java є ідеальним кандидатом для побудови об’єктно-орієнтованої архітектури, що була обрана на стадії проектування програмного комплексу. Java використовує автоматичний збирач сміття (garbage collector) для керування пам'яттю під час життєвого циклу об'єкта [28].

Віртуальна машина сканує на наявність непотрібних більше об'єктів. Коли до певного об'єкта вже не залишається посилань, збирач сміття може автоматично прибирати його із пам'яті. Проте, витік пам'яті все ж може статися, якщо код, написаний програмістом, має посилання на вже непотрібні об'єкти, наприклад на об'єкти, що зберігаються у діючих контейнерах.

Однією з важливих переваг платформи Java є наявність спеціальних класів для роботи з цілими числами довільної довжини – клас BigInteger. Даний клас в нашому програмному комплексі дозволяє реалізувати основні операції над елементами скінченного поля та ефективно представляти параметри еліптичної кривої.

Python – інтерпретована об'єктно-орієнтована мова програмування високого рівня з строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. В мові програмування Python підтримується такі парадигми програмування, як: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Основними її перевагами є:

– чистий та інтуїтивно зрозумілий синтаксис.

– портабельність програм пов’язана з інтерпретативнісю.

– наявність вбудованих модулів,шо надають можливість використовувати колекції, особливі структури даних, математичний пакет тощо.

– можливість програмування на Python в скрипковому режимі (з консолі).

– зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини).

– політика відкритого коду (open source).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості

платформ. Інтерпретатор мови Python і Стандартна бібліотека можуть вільно розповсюджуватись та використовуватись.

Інтерпретатор мови Python може бути розширений функціями та типами даних, розробленими на C або на іншій мові, яку можна викликати із C [29]. Python також зручна як мова розширення для прикладних програм, що потребують подальшого налагодження.

Описані вище особливості роблять Python привабливим кандидатом, що дозволив би розробити програмний комплекс для роботи з еліптичними кривими та дослідження методів багатократного скалярного множення точок еліптичної кривої на число.

З огляду на особливості обраних для порівняння мов програмування, можливості які вони надають, та потенційну ефективність розробки, було прийнято рішення обрати мову програмування Java як головний інструмент розробки програмного комплексу. Дана мова програмування надає можливість розробки якісної об’єктно-орієнтованої моделі, що ефективно моделює предметну галузь та надає для цього ряд стандартних засобів, як то класи для роботи з цілочисельними значеннями довільної довжини, стандартні контейнери об’єктів, наявність шаблонних типів, можливості написання високоефективних алгоритмів в процедурному стилі тощо.

3.4 Архітектура програмного комплексу

Розроблений програмний комплекс умовно можна розділити на три основні частини: модуль операцій в скінченних полях; модуль операцій над точками еліптичної кривої; модуль електронно-цифрового підпису. Також в рамках даної роботи розроблений модуль оцінки швидкодії виконання операції електронно-цифрового підпису в залежності від обраного методу оптимізації методу багатократного скалярного множення точки еліптичної кривої на число.

3.5 Модуль виконання операцій в скінченних полях

Модуль виконання в скінченних полях виконаний в об'єктно-орієнтованому стилі з використанням високого рівня абстракції – публічними є лише класи-абстракції, а саме FiniteField, FiniteFieldElement, FiniteFieldFactory, FiniteFieldAtithmetics, FiniteFieldElementIterator.

В той самий час, всі конкретні реалізації, а саме реалізації для скінченних полів $GF(p)$ та $GF(2^m)$ мають лише пакетну видимість.

Даний принцип приховування реалізації цілком задовольняє одні з основоположних принципів в об’єктно-орієнтованому моделюванні - SOLID принципи [30].

Під час проектування даного модуля були використані підходи описані нижче. Ієрархії класів, що моделюють скінченні поля зображено на рисунку 3.2 за допомогою діаграми класів мови моделювання UML.

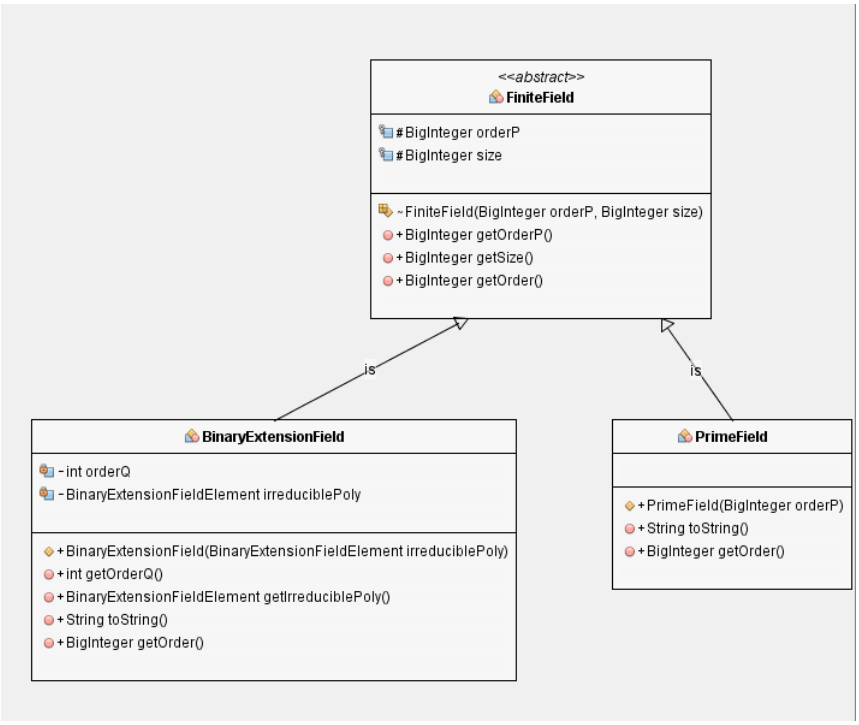


Рисунок 3.2 – Ієрархія класів, що моделюють скінченні поля

В залежності від типу скінченного поля –$GF(p)$ або $GF(2^m)$, елементи даного поля відрізняються у ряді своїх властивостей, а саме – відображення, тобто у вигляді числа або у вигляді многочлена, та способа зберігання у пам’яті, тобто одні зберігаються як звичайні цілочисельні значення, а інші мають додаткову характеристику – степінь многочлена.

Дану проблему в межах нашої роботи пропонується вирішувати за допомогою шаблона [31], принцип якого полягає у тому, що кожна конкретна реалізації інтерфейсу «фабрика» з єдиними методом «створити елемент» по різному його реалізує, а саме створює деякий клас-потомок із ієрархії наслідування класу «Елемент скінченного поля».

Графічно даний підхід представлений на рисунку 3.3 за допомогою діаграми класів мови моделювання UML.

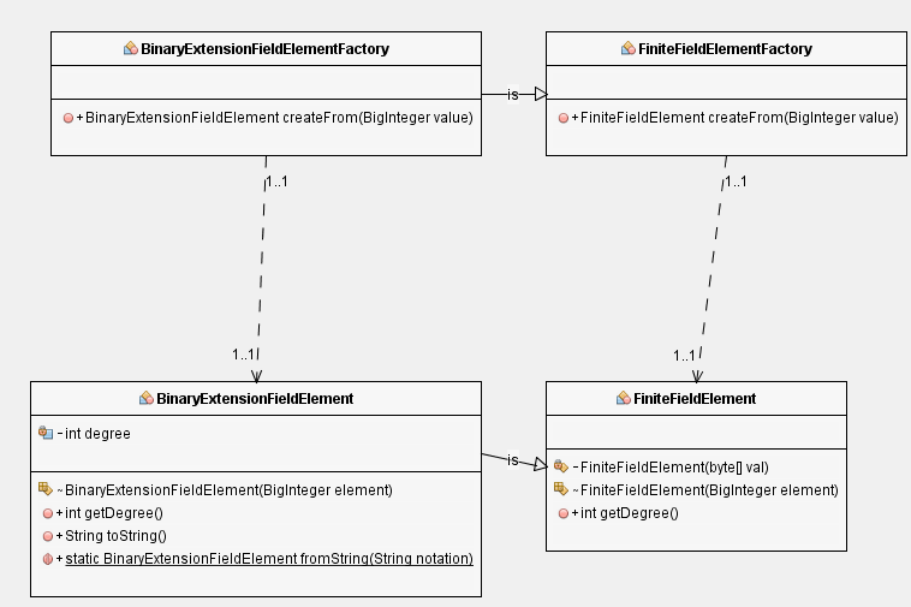


Рисунок 3.3 – Модель створення конкретного елемента скінченного поля в залежності від його типу

Ітерація елементів скінченного поля за допомогою шаблона

Враховуючи той факт, що скінченні поля будуються за певними математичними принципами, описаними докладно в розділі 2 даної роботи, було би недоцільно виділяти сталий об’єм пам’яті для зберігання елементів поля, наприклад використовуючи колекції, для вирішення даної проблеми був застосований шаблон[32], що дозволяє перебирати елементи деякої колекції, не знаючи її розмірів. В результаті, під час ітерація, елементи поля вираховуються базуючись на поточному елементі ітерація. Це дозволяє обходити елементи скінченного поля, динамічно ініціалізуючи їх у пам’яті. Більше того, невикористовувані елементи-об’єкти в пам’яті будуть видалені під час наступного проходу.

Графічно даний підхід представлений на рисунку 3.4 за допомогою діаграми класів мови моделювання UML.

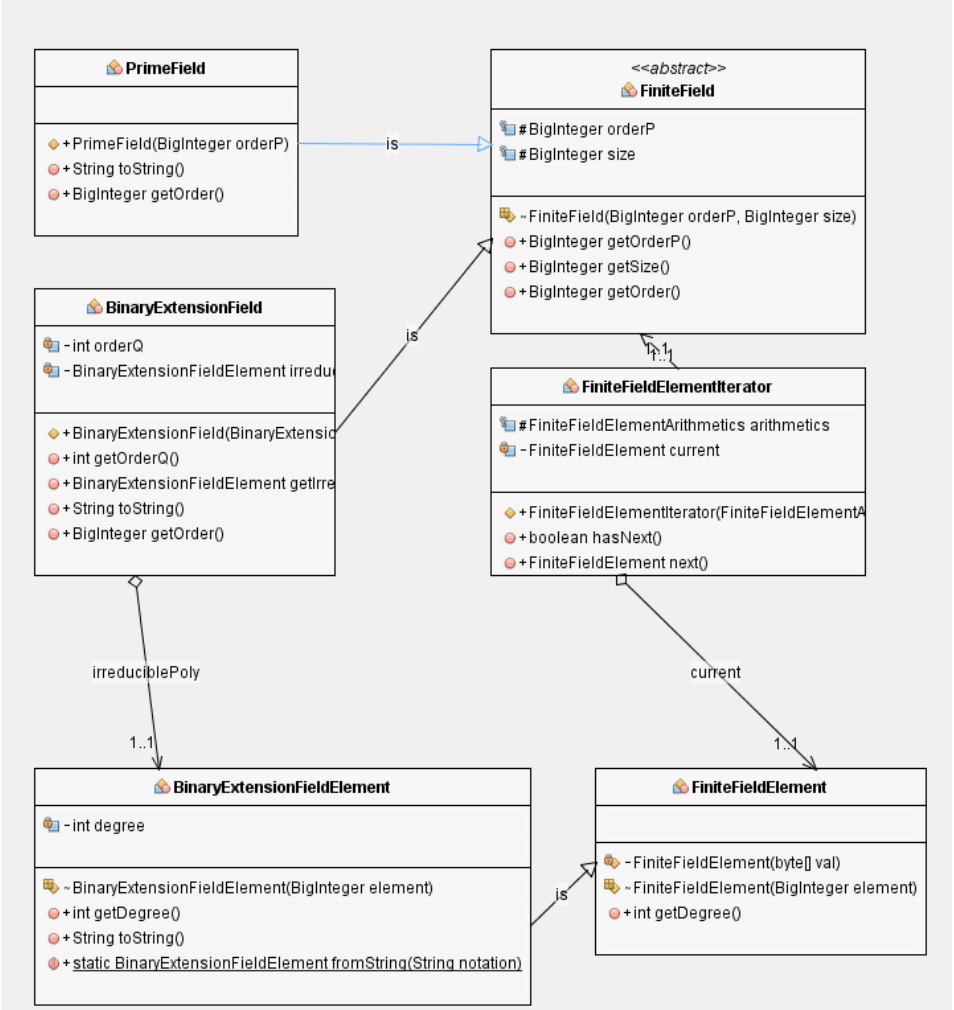


Рисунок 3.4 – Модель ітерації елементів скінченного поля

3.6 Інкапсуляція виконання арифметичних операцій над елементами скінченного поля

Для виконання арифметичних операцій над елементами скінченного поля таких як додавання, віднімання, множення елементів, а також обчислення оберненого елемента було прийнято рішення виділити абстракцію, що відповідає за виконання операцій. Даний підхід дозволяє інкапсулювати конкретні реалізації алгоритмів, що використовуються в процесі арифметичних обчислень, залишаючи можливість для створення класів-нащадків, що містили б різноманітні оптимізації для покращень характеристик даних обчислень.

Графічно даний підхід представлений на рисунку 3.5 за допомогою діаграми класів мови моделювання UML.

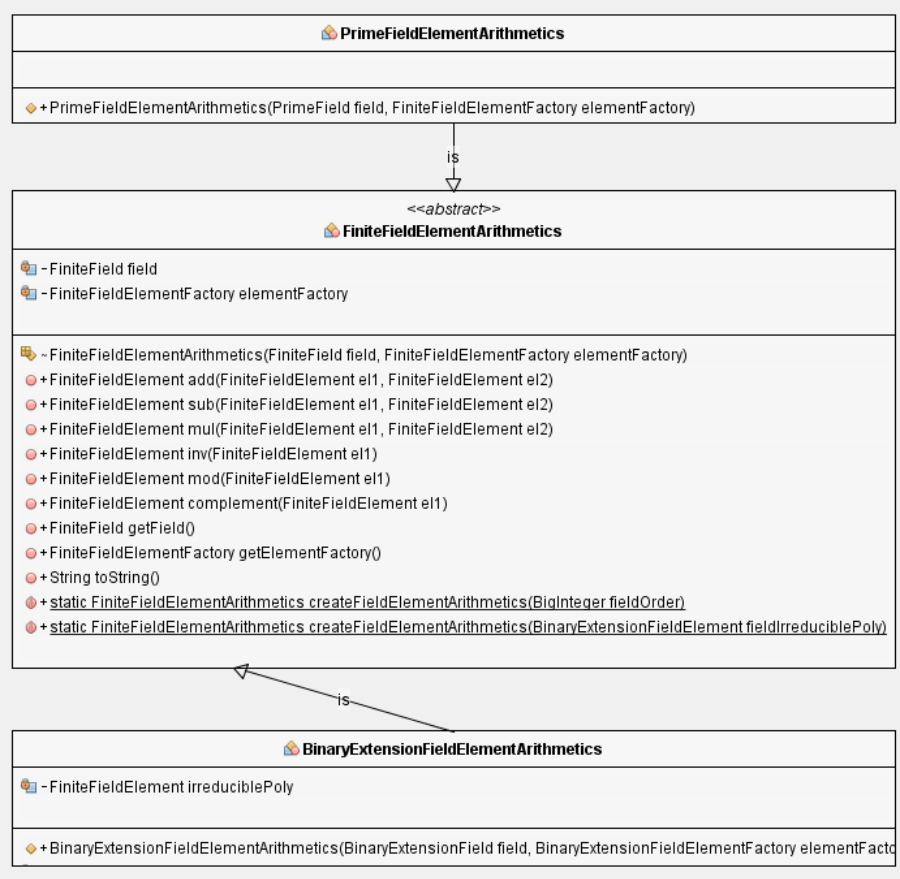


Рисунок 3.5 – Модель інкапсуляції виконання арифметичних операцій над елементами скінченного поля

3.7 Модуль виконання операцій над точками еліптичної кривої

Як було зазначено у попередніх розділах, в межах даної роботи особливу увагу приділяється дослідженню еліптичних кривих з параметрами, визначеними саме над скінченними полями $GF(p)$ та $GF(2^m)$. Модуль операцій над точками еліптичної кривої нерозривно пов'язаний з модулем виконання операцій над елементами скінченного поля. З іншого боку, за принципом мінімальної поінформованості в об'єктно-орієнтованій архітектурі програмного забезпечення, необхідно прагнути до зменшення залежності між модулями програмного комплексу.

У розробленому програмному комплексі для виконання операцій над точками еліптичної кривої, за рахунок прийнятих архітектурних рішень, описаних в даному розділі, вдалося досягти залежності лише від однієї абстракції – класу реалізації арифметики над елементами скінченного поля. Всі інші елементи даного модуля

використовуються неявно, а саме в тілі конкретних реалізацій класів арифметики. Таким чином розроблений модуль задовольняє вимогу мінімальної поінформованості.

3.8 Інкапсуляція виконання арифметичних операцій над точками еліптичної кривої

Аналогічно до принципу інкапсуляції операцій над елементами скінченного поля, з огляду на принцип об'єктно-орієнтованого проектування, що стверджує, що класи повинні бути закритими до модифікації але відкритими для розширення, було прийнято рішення також інкапсулювати арифметичні операції над точками еліптичної кривої. Графічно даний підхід представлений на рисунку 3.6 за допомогою діаграми класів мови моделювання UML.

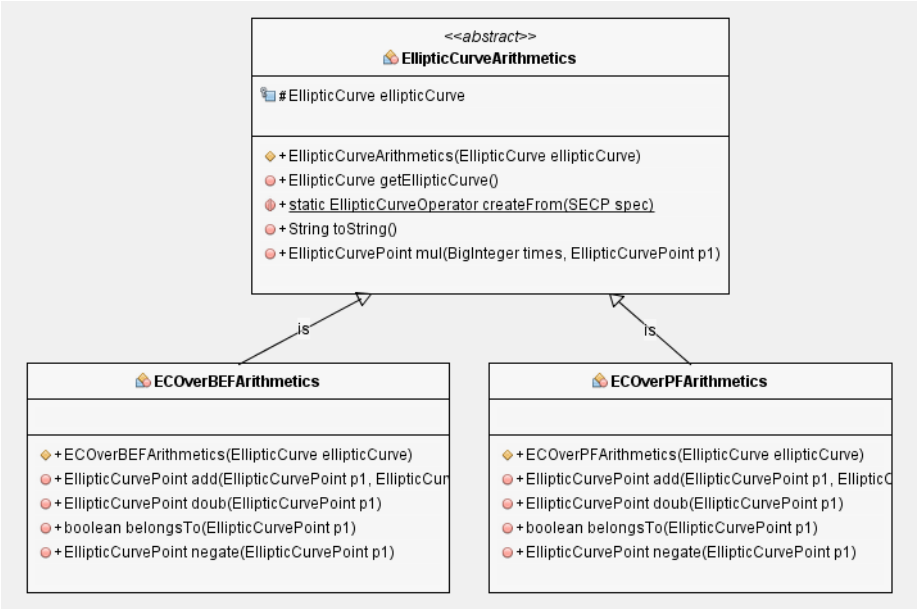


Рисунок 3.6 – Модель інкапсуляції виконання арифметичних операцій над елементами скінченного поля

3.9 Обробка граничних випадків, пов'язаних з операціями з точкою на нескінченності

Одними із граничних випадків (corner cases) під час виконання операцій над точками еліптичної кривої є операції за участю точки на нескінченності. Детальніше дана проблема була розглянута в розділі 2. В контексті розробки програмного комплексу для

виконання операцій з еліптичними кривими, складність реалізації даної функціональності пов’язана з тим, що еліптична точка на нескінченності не може бути виражена у вигляді цілочисельних координат, адже вважається, що вона знаходиться на нескінченності.

Для вирішення даної проблеми було використано ряд підходів, що забезпечили ефективне подолання даної колізії, зокрема декларування статичного члена класу точки еліптичної кривої, що грає роль точки на нескінченності та використання шаблона «Декоратор» [33] для попередньої перевірки вхідних параметрів методів, що приймають об’єкти-точки еліптичної кривої в якості аргументів.

Даний підхід дозволяє логічно розділити граничні випадки від самих алгоритмів, не порушуючи при цьому ієрархію класів, адже розроблений клас-декоратор має конструктор, що приймає клас арифметики, а також і сам наслідує даний клас. Графічно даний підхід представлений на рисунку 3.7 та рисунку 3.8 за допомогою діаграми класів мови моделювання UML.

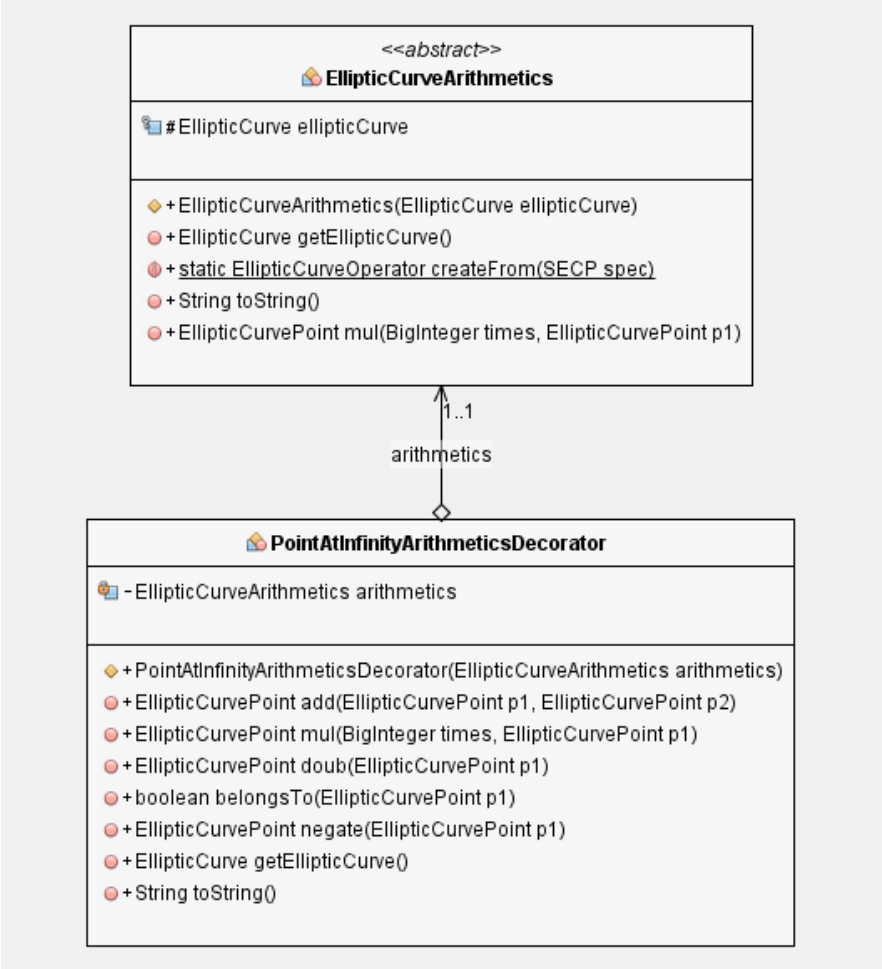


Рисунок 3.7 – Обробка крайніх випадків, пов’язаних з операціями з точкою на нескінченності

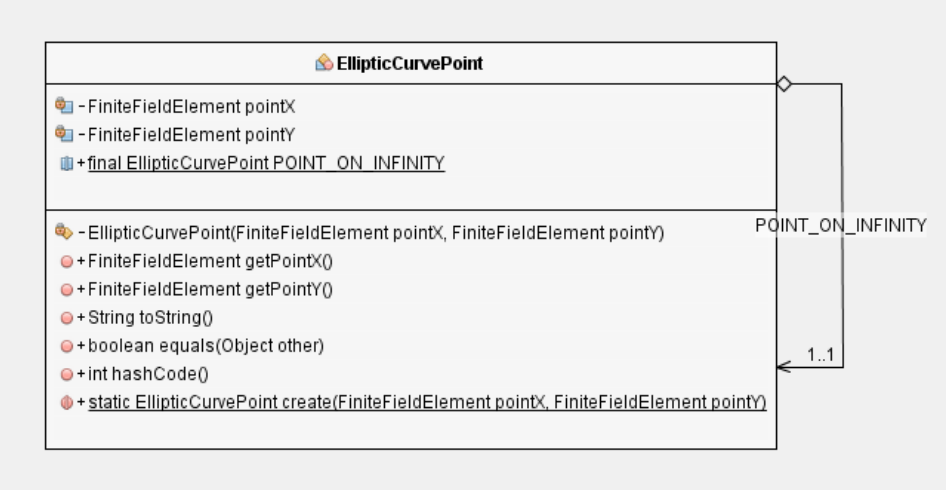


Рисунок 3.8 – Визначення точки на нескінченності за допомогою статичного поля

3.10 Визначення стандартних параметрів еліптичної кривої

За наявності стандарту параметрів еліптичної кривої, що забезпечують найкращу криптостійкість, а саме стандарту SECP, в процесі розробки програмного комплексу постала необхідність їх використання в тілі модуля. З огляду на те, що кількість параметрів еліптичної кривої є однаковою незалежно від типу скінченного поля, над яким вона визначена, можливість використання перечислення стало найбільш раціональним способом їх об’явлення в тілі тексту програми.

Окрім цього, з використанням даного підходу, ефективним способом створення об’єктів еліптичної кривої є використання статичного шаблонного методу. Даний підхід, на відміну від використання конструктора, дозволяє повертати класи-нащадки, тобто класи, що залежать від об’єктів, обраних в залежності від типу скінченного поля, над яким визначається дана крива. Графічно даний підхід представлений на рисунку 3.9 та рисунку 3.10 за допомогою діаграми класів мови моделювання UML.

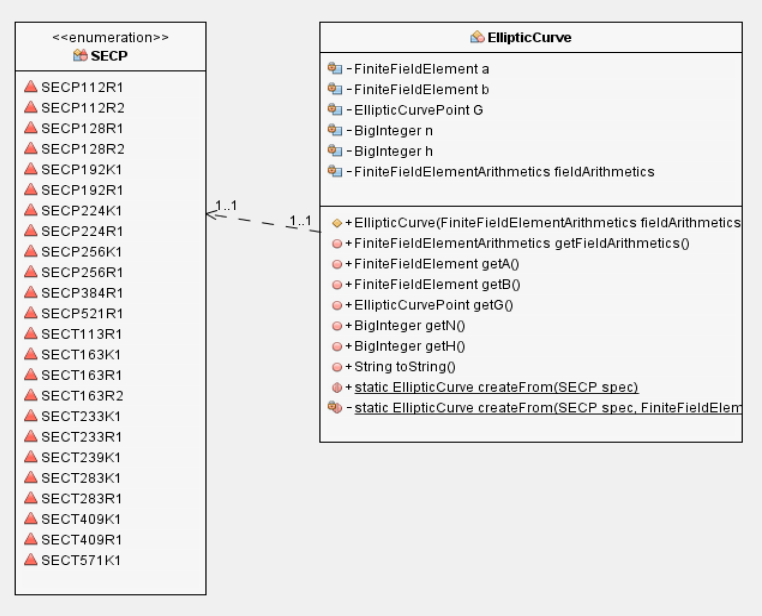


Рисунок 3.9 – Використання стандартних кривих SECP

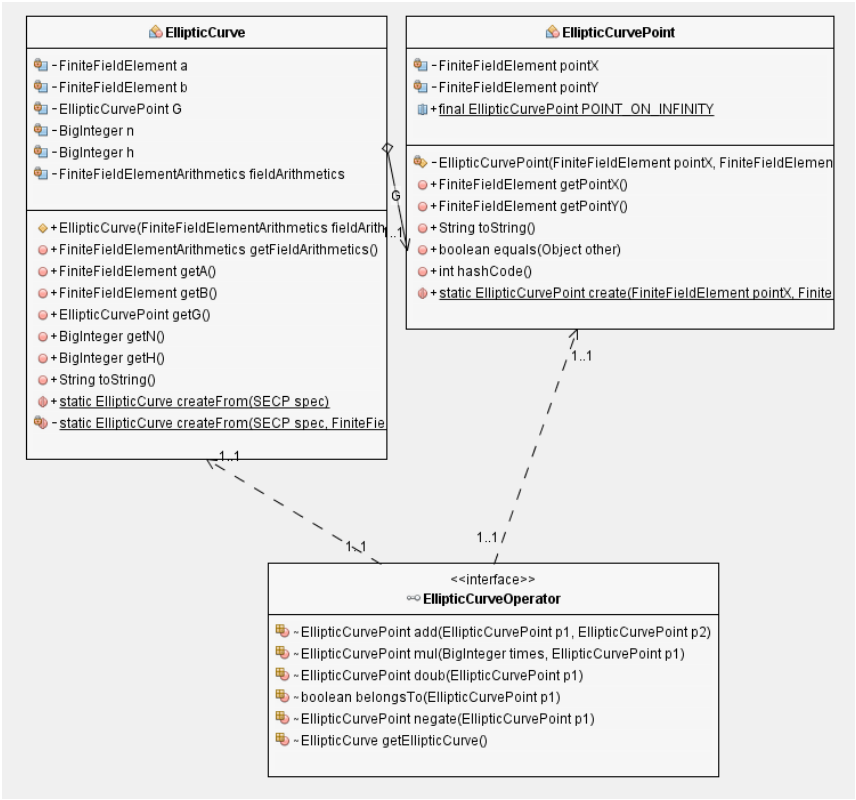


Рисунок 3.10 – Використання статичного шаблонного методу, що приймає стандарт SECP

3.11 Модуль електронно-цифрового підпису

Враховуючи наявність модуля виконання операцій над елементами скінченного поля та модуля виконання операцій над точками еліптичної кривої, модуль електронно-цифрового підпису найбільш високорівневим інтерфейсом доступу до розробленого програмного комплексу та його головною частиною. Публічний інтерфейс даного модуля представлений абстракцією, що презентує виконання підпису та перевірки ЕЦП та абстракцією пари ключів криптографічної схеми ECDSA.

Основний інтерес в даному модулі представляє спосіб організації класу, що реалізує абстракцію електронно-цифрового підпису, що дозволяв би інкапсулювати алгоритм багатократного скалярного множення точки еліптичної кривої на число. Теорія об'єктно-орієнтованого проектування містить класичний підхід до вирішення такого типу проблем, а саме – шаблон проектування [35].

Даний шаблон, з огляду на принцип об'єктно-орієнтованого проектування, що проголошує пріоритет композиції над наслідуванням, дозволяє динамічно замінювати конкретні реалізації алгоритму, не змінюючи при цьому тіло класу, у якому викликається даний алгоритм. Конкретно в даному програмному комплексі, були реалізовані різноманітні стратегії методів багатократного скалярного множення точки еліптичної кривої на число, розглянутих в межах нашого дослідження, а саме – методи SMPM, JSF, NAF, DBNS, що були детально описані в розділі 3.

Графічно даний підхід представлений на рисунку 3.11 – за допомогою діаграми класів мови моделювання UML.

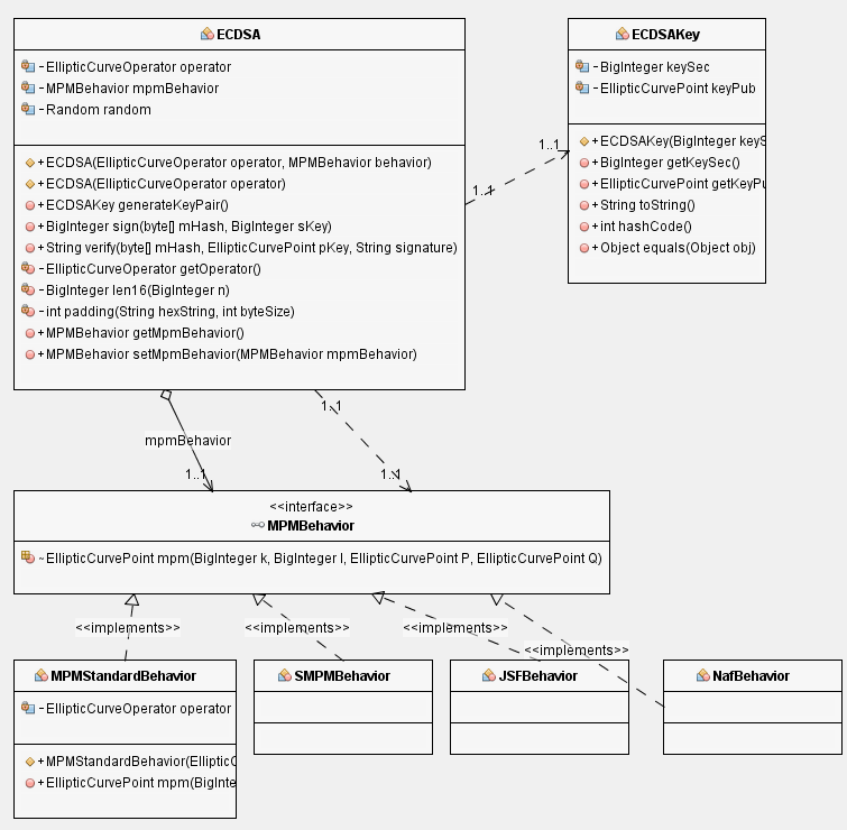


Рисунок 3.11 – Використання шаблону в модулі електронно-цифрового підпису

3.12 Програмна реалізація отримання цифрового підпису на основі алгоритму ECDSA

Програмний комплекс, реалізований на мові програмування C++ в середовищі Microsoft Visual Studio Net 2017 (додаток А), надає можливість отримання цифрового підпису на основі алгоритмів RSA, DSA, ECDSA та Ель Гамаля. Розмір програмного файлу *ESign.exe* – 88 Кбайт.

Підготовчі операції для отримання ЕЦП складаються з декількох етапів:

1. Запуск програми *ESign.exe*. Після запуску програми отримуємо робоче вікно програми (рис. 3.12). У лівому верхньому куті програми знаходяться кнопки керування програмою, які при наведені на них видають підказку щодо можливих дій.

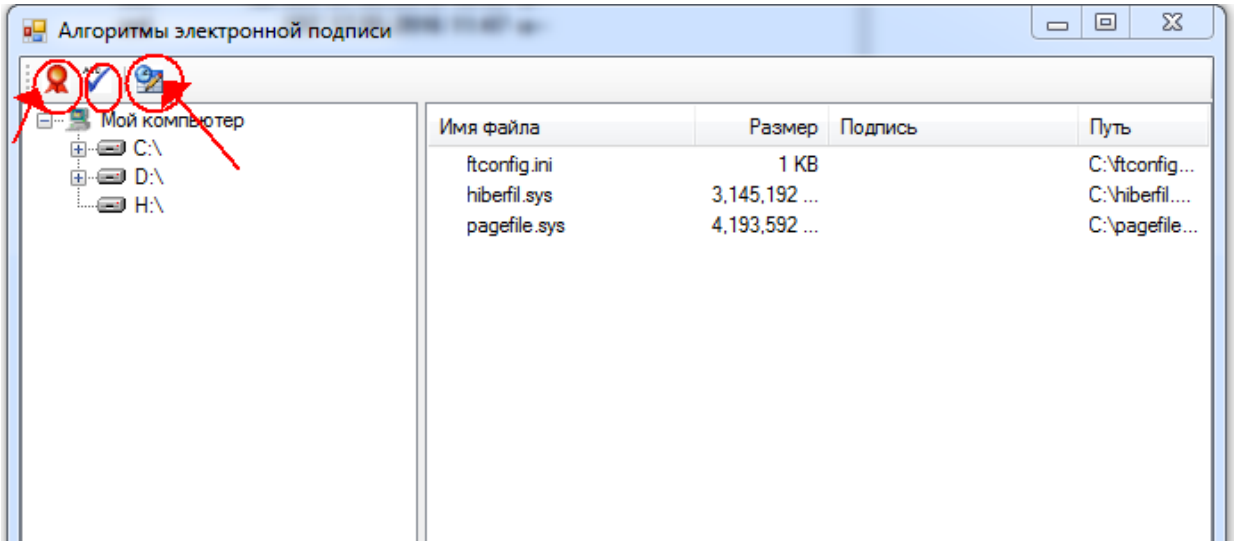


Рисунок 3.12 – Загальне вікно програми

2. Генерація закритого та відкритого ключів. На цьому етапі генерується закритий (key1.keysPr – приватний) та відкритий ключ (key1.keysPb – публічний). Для цього необхідно на головній формі натиснути кнопку «Налаштування даних» (рис. 3.13).

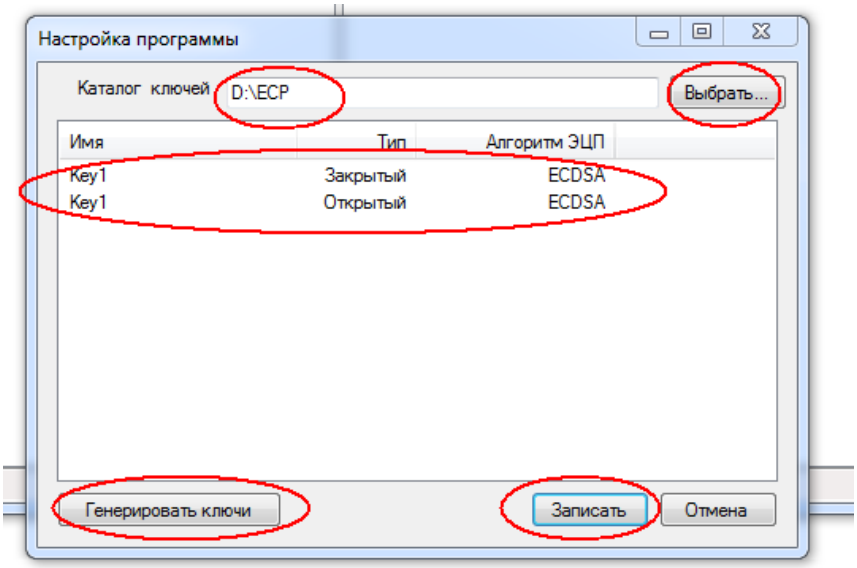


Рисунок 3.13 – Процес генерації відкритого та закритого ключів

Створені ключі (закритий і відкритий) зберігаються одній директорії (рис. 3.14). У даному прикладі це директорія ECP диску D.

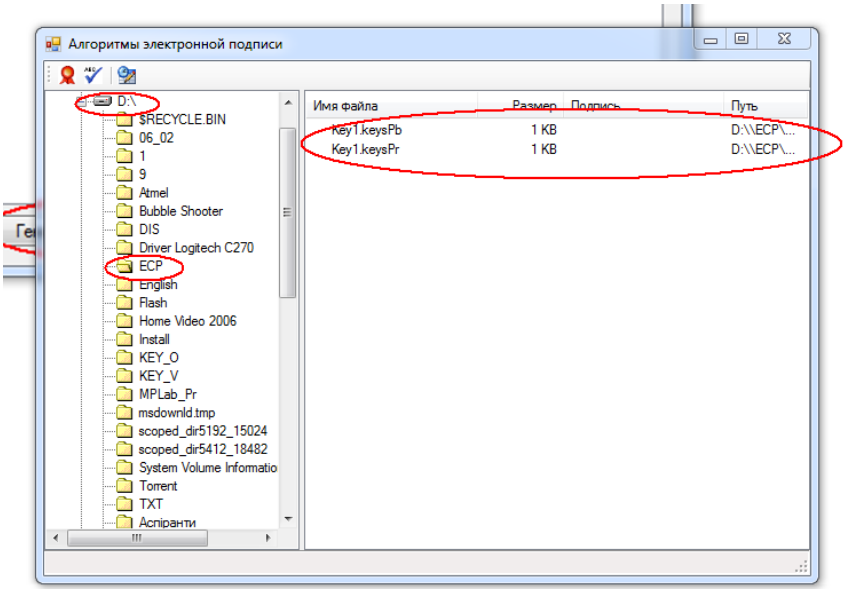


Рисунок 3.14 – Збереження закритого та відкритого ключів

3. Вибір алгоритму ЕЦП та файлу для підпису. У даному прикладі це файл «Рейтинг здобувачів вищої освіти.xlsx» розміром 49 кБ (рис. 3.15).

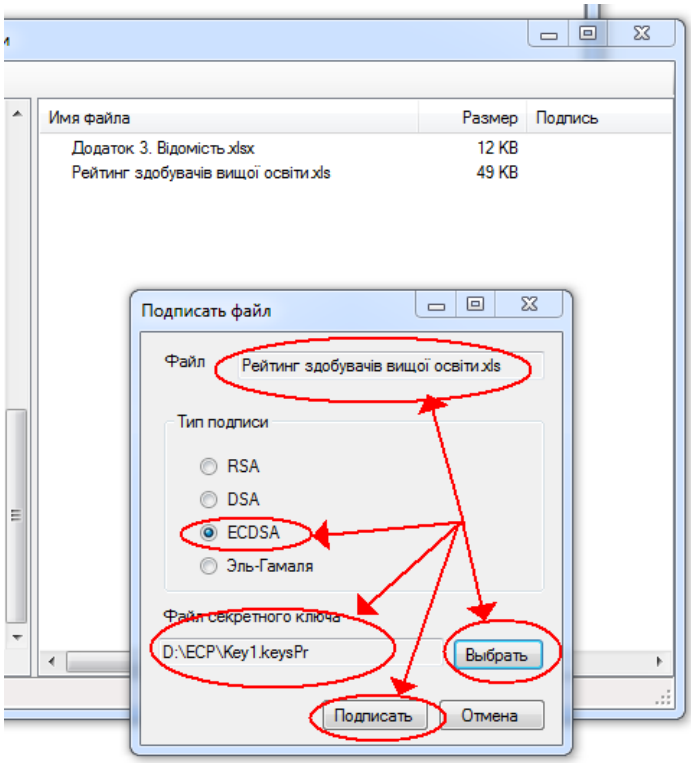


Рисунок 3.15 – Вибір алгоритму ECDSA та файлу для підпису

4. Вибір параметрів підпису. Для виконання цього етапу необхідно скористатись п. 3.2 даної роботи та обрати параметри для підпису, згідно рекомендацій NIST.

NIST рекомендує 15 еліптичних кривих, багато з яких були отримані Jerry Solinas (NSA) на базі напрацювань Neal Koblitz.

Зокрема, FIPS 186-3 (Federal Information Processing Standards) рекомендує 10 кінцевих полів, що мають довжину 192, 224, 256, 384 або 521біт.

Параметр NIST у програмі, згідно рекомендацій, може приймати значення 192 (за замовчуванням), 224, 256, 384 або 521 біт.

Обрані параметри підпису наведені на рисунку 3.16.

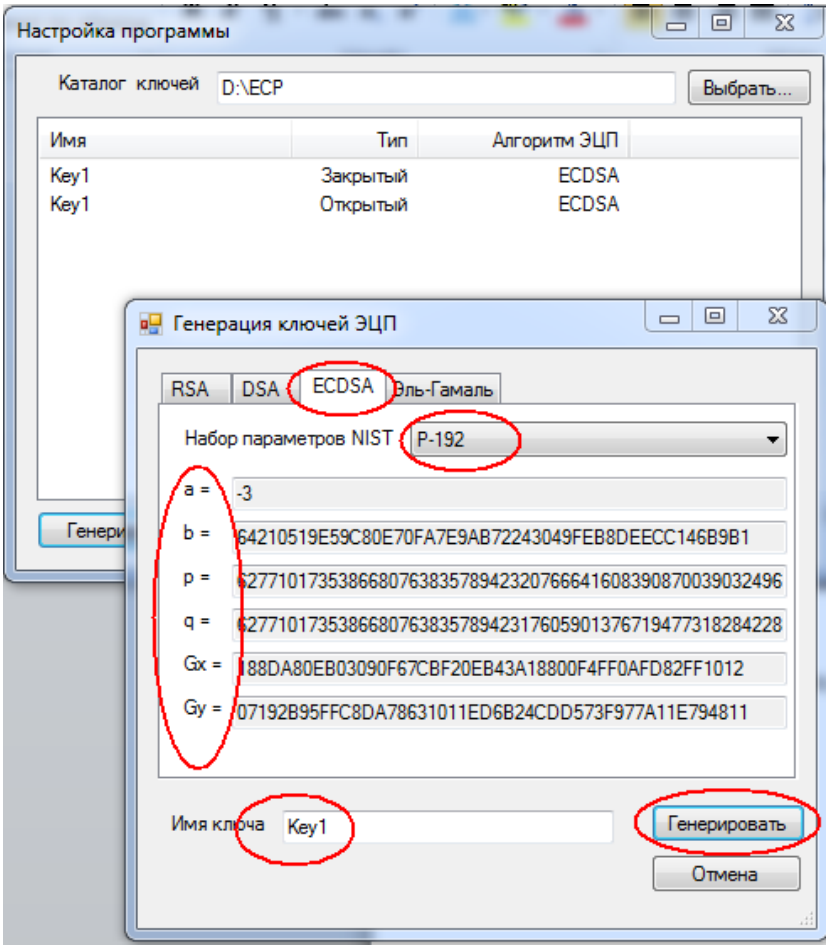


Рисунок 3.16 – Вибір параметрів алгоритму ECDSA

5. Збереження файлу, підписаного ЕЦП. Після підпису створюється файл з розширенням *.sign* та він записується в обрану директорію (рис. 3.17).

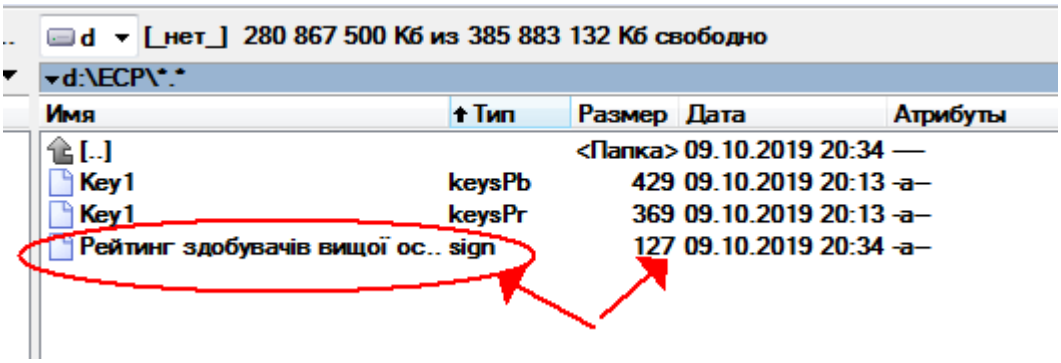


Рисунок 3.17 – Запис файлу з ЕЦП

Як видно з рисунка 3.17 розмір файлу збільшився, оскільки відбулося приєднання ЕЦП.

6. Етап перевірки ЕЦП. Для виконання перевірки ЕЦП на головному екрані меню програми слід обрати файл тексту з ЕЦП, файл відкритого ключа, файл ЕЦП та натиснути кнопку «Перевірити підпис» (рис. 3.18).

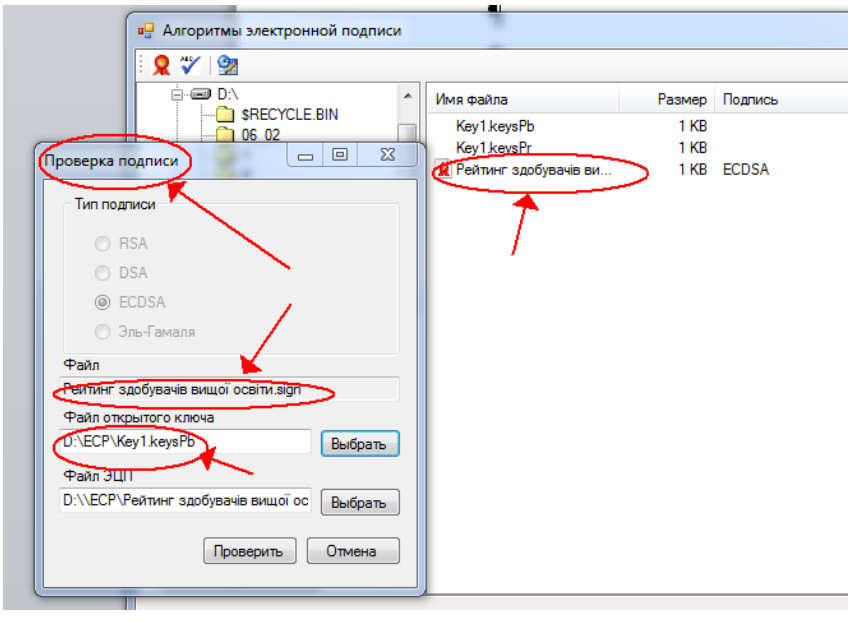


Рисунок 3.18 – Вибір параметрів для перевірки ЕЦП

При правильності ЕЦП програма видає повідомлення «ОК», що свідчить про відповідність ЕЦП оригіналу (рис. 3.19). В іншому випадку програма видає помилку.

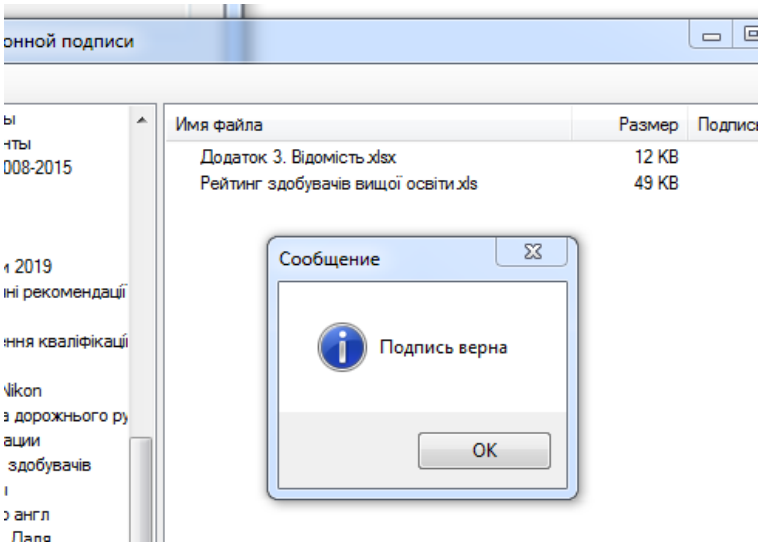


Рисунок 3.19– Завершення перевірки ЕЦП

Програма здійснює шифрування з обраними параметрами та зберігає ЕЦП на диску або зовнішньому носію. Програма володіє загально прийнятим інтуїтивно зрозумілим інтерфейсом та підписаними опціями, за замовчуванням сама обирає необхідне розширення файлу, що спрощує роботу з нею користувачів, що не володіють достатніми навиками роботи на персональному комп’ютері.

3.13 Висновки до розділу 3

В даному розділі були розглянуті підходи, що використовувались на стадії проектування програмного комплексу для дослідження та оптимізації методу багатократного скалярного множення точки еліптичної кривої на число.

Дані підходи, а саме використання шаблону для інкапсуляції обробки виключних ситуацій, пов’язаних з колізіями точки на нескінченності, шаблону для реалізації неявнового виклику на створення непублічних субкласів, шаблону для інкапсуляції різноманітних методів оптимізації багатократного скалярного множення точки еліптичної кривої на число тощо, дозволили виконати архітектуру програмного комплексу з дотриманням кращих стандартів об’єктно-орієнтованого проектування, а саме принципів SOLID та інших.

3.14 Перелік посилань до розділу 1-3

1. Darrel Hankerson, Alfred Menezes, Scott Vanstone Guide to Elliptic Curve Cryptography / Darrel Hankerson // 2004, Springer – Verlag New York Inc. – 205 p.

2. Мао, Венбао. Современная криптография: теория и практика. : Пер. с англ. — М. : Издательский дом «Вильямс», 2005. — 768 с.

3. N.Koblitz, Elliptic Curve Cryptosystems [Text]/ Neal Koblitz// Mathematics of Computation – Volume 48 – Number 177 – January 1987 – pp. 203-209

4. Cetin Kaya Coc, Cryptographic Engineering/ Cetin Kaya Coc// Springer Science+Business Media, LLC 2009 – 171 p

5. Василенко О. Н., Теоретико-числовые алгоритмы в криптографии/ Василенко О. Н.// М.: МЦНМО, 2003. – 178 с.

6. J. Solinas. Low-weight binary representations for pairs of integers/ J. Solinas// National Security Agency, USA 2001.

7. V. S. Dimitrov, L. Imbert, and P. Mishra. Fast elliptic curve point multiplication using double-base chains. 2005.

8. Сидельников, В.М. Криптография и теория кодирования [Текст] / В.М. Сидельников. — М. : ФИЗМАТЛИТ, 2008. — 324 с.

9. Черемушкин, А.В. Лекции по арифметическим алгоритмам в криптгографии. — М. : МЦНМО, 2002. — 104 с.

10. Панасенко, С.П. Алгоритмы Шифрования. Специальный Справочник [Текст] / С.П. Панасенко. — М. : Академический Проект, 2006. — 529 с.

11. Василенко, О.Н. Теоретико-числовые алгоритмы в криптографии [Текст] / О.Н. Василенко. — М. : МЦНМО, 2003. — 328с.

12. Joye, M. Cryptographic Hardware and Embedded Systems — CHES 2004 [Text] / Marc Joye. — Cambridge, USA : Springer, 2004. — 471 p.

13. Фергюсон. Практическая криптография: Пер. с англ. [Текст] / Нильс, Шнайер, Брюс. — М.: «Вильямс», 2005. — 424 с.

14. Князькіна, О.С. Спосіб генерування випадкового простого числа заданої довжини [Текст] / М.В. Онай, О.С. Князькіна // Праці міжнародн. конф «СНКПМІ-2012». — 2012. — С. 52-54.

15. Ярочкин, В.И. Информационная безопасность: Учебник для вузов [Текст] / В.И. Ярочкин. — М. : Академический Проект, 2008. — 544 с.

16. W. Diffie and M. Hellman New directions in cryptography. Information Theory, IEEE

Transactions on elliptic Curves, 22 : 644-654, 1976.

17. N. Koblitz Elliptic curve cryptosystems. Mathematics of Computation, 48 : 203-209, 1987.

18. V.S. Miller Use of Elliptic Curves in Cryptography, page 417, 1986.

19. D.Hankerson, A.Menezes, S.Vanstone Guide to Elliptic Curve Cryptography, page 311, 2004.

20. Jithra Adikari, Vasil S.Dimitrov and Pradeep Mishra Fast Multiple Point Multiplication on Elliptic Curves over Prime and Binary Fields using the Double-Base Number System.

21. Mathiew Ciet and Marc Joye Trading Inversions for Multiplications in Elliptic Curve Cryptography

22. Vasyl Dimitrov, Laurent Imbert and Pradeep Kumar Mishra Efficient and Secure Elliptic Curve Point Multiplication Using Double-Base Chains

23. Duc-Phong Le, Bin P.Nguyen Fast Point Quadrupling on Elliptic Curves

24. Gordon H. Bredley. Algorithm and bound for the greatest common divisor of n integers. Communications of the ACM, 13(7):433-436, 1970.

25. Tudor Jebelean. A generalization of the binary GCD algorithm. In M. Bronstein, editor, 1993 ACM International Symposium on Symbolic and Algebraic Computation, pages 111—116, Kiev, Ukraine, 1993. ACM Press.

26. D. E. Knuth. The Art of Computer Programming: Seminumerical Algorithms, volume 2. Addison-Wesley, Reading, Mass., 2nd edition, 1981.

27. D. H. Lehmer. Euclid's algorithm for large numbers. Amer. Math. Monthly, 45:227-233, 1938.

28. G. Norton. Extending the binary GCD algorithm. In J. Calmet, editor, Proceedings of the 3rd International Conference on Applied Algebra, pages 363-372. Springer-Verlag, 1985. LNCS 229.

29. J. M. Pollard. Theorems on factorization and primality testing. Proc. Camb. Phil. Soc., 76:521-528, 1974.

30. Jonathan P. Sorenson. Two fast GCD algorithms. Journal of Algorithms, 16:110-144, 1994.

31. Jonathan P. Sorenson. An analysis of Lehmer's Euclidean GCD algorithm. In

A. H. M. Levelt, editor, 1995 ACM International Symposium on Symbolic and Algebraic Computation, pages 254-258, Montreal, Canada, July 1995.

32. Князькіна, О.С. Алгоритми виконання низькорівневих операцій на еліптичній кривій [Текст] / М.В. Онай, О.С. Князькіна // Міжнародна науково-практична конференція «Проблеми інформатики та комп'ютерної техніки». Праці конференції. — Чернівці:

Видавничий дім "Родовід", 2014. — С. 87 - 89.

33. Дичка А.І. Модифікація методу багатократного скалярного множення точок еліптичної кривої на число [Текст] / М.В. Онай, Дичка А.І. // Прикладна математика та комп'ютинг. ПМК, 2014 : десята наук. конф. магістрантів та аспірантів, Київ, 16-18 квітня 2014 р. : зб. тез доп. / [ред кол.: Дичка І.А. та ін.] . – К. : Просвіта, 2018. – С. 235-240.

34. Дичка А.І. Дослідження швидкодії методів вирішення задачі дискретного логарифма на еліптичних кривих [Текст] / М.В. Онай, О.С. Дичка А.І. // Праці міжнародної науково-практичної конференції «Інформаційні технології та комп'ютерна інженерія». — 2017. — С. 172- 176.

35. Knyazkina, O.S. Finding of reverse element in the multiplicative group of the ring residues Z/mZ [Text] / M.V. Onay, O.S. Knyazkina // Праці міжнародн. конф. «Імені академіка М. Кравчука». Том 2. — К. : НТУУ «КПІ». — 2012. — С. 23-24.

36. Value Proposition Canvas Template [Електронний ресурс]. – дата візиту 04.12.2019. – Режим доступу до ресурсу: https://goo.gl/MbhgG4

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даної роботи магістра було провести дослідження алгоритму створення електронного цифрового підпису з використанням групи точок еліптичної кривої , і як результат було створено повноцінний проект, котрий можна використовувати у розробці такої системи. Так як в процесі проектування використовувалося спеціалізоване приміщення, то аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера на якому буде розроблятися розроблена система. При роботі з обчислювальною технікою змінюються фізичні і хімічні фактори навколишнього середовища: виникає статична електрика, електромагнітне випромінювання, змінюється температура і вологість, рівень вмісту кисню й озону в повітрі. Також недотримання вимог безпеки призводить до того, що при роботі за комп'ютером працівник може відчувати дискомфорт: виникають головні болі й різь в очах, з'являються втома й дратівливість. У деяких людей порушується сон, апетит, погіршується зір, починають хворіти руки, шия, поперек тощо. При ненормованій роботі можливе нервове виснаження. Забезпечення цих умов покладається на власника або уповноважений ним орган (далі роботодавець). Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці, що чітко врегульовані законодавством України .

4.1 Аналіз стану умов праці

Робочі місця офісних працівників, обладнані персональними комп’ютерами (далі – робочі місця), повинні відповідати вимогам «Правил охорони праці під час експлуатації електронно-обчислювальних машин», затверджених Наказом Державного комітету України з промислової безпеки, охорони праці та гірничого нагляду від 26.03.2010 року № 65 (Правила), та «Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин», затверджених постановою Головного

державного санітарного лікаря України від 10.12.98 N 7 (ДСанПіН 3.3.2-007-98 [1]). Правила поширюються на всіх суб'єктів господарювання незалежно від форм власності, які у своїй діяльності здійснюють роботу, пов'язану з персональними комп'ютерами, у тому числі на тих, які мають робочі місця, обладнані персональними комп'ютерами і периферійними пристроями. Зазначені нормативно-правові акти встановлюють санітарно-гігієнічні вимоги до приміщення, в якому розташоване робоче місце, власне до робочого місця, освітлення, рівнів вібрації і шуму, мікроклімату в приміщенні тощо.

Приміщення

Будівлі та приміщення, де розміщені робочі місця, повинні відповідати вимогам нормативно-технічної та експлуатаційної документації виробника персональних комп'ютерів ДСанПіН 3.3.2-007-98 [1] та Правил. Будівлі та приміщення, де розміщені робочі місця операторів, мають бути не нижче другого ступеня вогнестійкості. Для всіх будівель і приміщень, де знаходяться робочі місця, повинно бути визначено клас зони згідно з НПАОП 40.1-1.01-97 [2]. Відповідне позначення повинно бути нанесено на вхідних дверях кожного приміщення. Не дозволяється розташування приміщень з робочими місцями у підвалах і цокольних поверхах. Неприпустимим є розташування приміщень категорій А і Б, а також виробництв з мокрими технологічними процесами поряд з приміщеннями, де розташовуються робочі місця, а також над ними чи під ними. При цьому площа приміщення має бути не менше 6,0 кв. м. із розрахунку на одне робоче місце, а об'єм – не менше 20,0 куб. м.

Віконні прорізи приміщень для роботи з персональними комп'ютерами мають бути обладнані регульованими пристроями (жалюзі, завіски, зовнішні козирки. Для внутрішнього оздоблення приміщень з персональними комп'ютерами слід використовувати дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі 0,7-0,8, для стін 0,5-0,6. Покриття підлоги повинне бути матовим з коефіцієнтом відбиття 0,3-0,5. Поверхня підлоги має бути рівною, неслизькою, з антистатичними властивостями. Забороняється для оздоблення інтер'єру приміщень з персональними комп'ютерами застосовувати полімерні матеріали (деревинно-стружкові плити, шпалери, що миються, рулонні синтетичні матеріали, шаруватий паперовий пластик тощо), що виділяють у повітря шкідливі хімічні речовини. Полімерні матеріали для внутрішнього оздоблення приміщень з персональними комп'ютерами можуть бути використані при наявності дозволу органів та установ державної санітарно-епідеміологічної служби. Приміщення можуть обладнуватись шафами для зберігання документів, магнітних дисків, полицями, стелажами, тумбами тощо з урахуванням вимог до площі приміщень.

У приміщеннях з джерелами шкідливих виробничих факторів робочі місця операторів мають розміщуватися в ізольованих кабінах, які обладнані повітрообміном.

Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном), мають бути надійно захищені діелектричними щитками або сітками з метою недопущення потрапляння працівника під напругу. Приміщення, де розміщені робочі місця, мають бути оснащені системою автоматичної пожежної сигналізації і вогнегасниками відповідно до вимог чинного законодавства України. Проходи до засобів пожежогасіння мають бути вільними.

У приміщеннях, в яких розташовані робочі місця, слід щоденно робити вологе прибирання. Крім того, ці приміщення мають бути оснащені аптечками першої медичної допомоги, а при них мають бути обладнані побутові приміщення для відпочинку під час роботи, кімната психологічного розвантаження.

Організація та обладнання робочого місця

При розміщенні робочих столів з персональними комп'ютерами слід дотримувати:

- відстань між бічними поверхнями персональних комп'ютерів 1,2 м.;
- відстань від тильної поверхні одного персонального комп'ютера до екрана іншого – 2,5 м.

За потреби особливої концентрації уваги під час виконання робіт суміжні робочі місця операторів необхідно відділяти одне від одного перегородками висотою 1,5 – 2м.

Конструкція робочого місця користувача персонального комп'ютера має забезпечити підтримання оптимальної робочої пози офісного працівника. Конструкція робочого столу має відповідати сучасним вимогам ергономіки і забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання (дисплея, клавіатури, принтера) і документів. Висота робочої поверхні робочого столу має регулюватися в межах 680-800 мм, а ширина і глибина – забезпечувати можливість виконання операцій у зоні досяжності моторного поля (рекомендовані розміри: 600-1400мм, глибина – 800-1000мм).

Робочий стіл повинен мати простір для ніг заввишки не менше ніж 600мм, завширшки не менше ніж 500мм, завглибшки (на рівні колін) не менше ніж 450мм, на рівні простягнутої ноги не менше ніж 650мм. Робочий стілець має бути підйомно-поворотним, регульованим за висотою, з кутом і нахилу сидіння та спинки і за відстанню від спинки до переднього краю сидіння поверхня сидіння має бути плоскою, передній край – заокругленим. Регулювання за кожним із параметрів має здійснюватися незалежно, легко і надійно фіксуватися. Шаг регулювання елементів стільця має становити: для

лінійних розмірів – 15-20мм, для кутових – 2-5 градусів. Зусилля регулювання має не перевищувати 20Н. Висота поверхні сидіння має регулюватися в межах 400-500мм, а ширина і глибина становити не менше ніж 400мм. Кут нахилу сидіння – до 15 градусів вперед і до 5 градусів назад. Висота спинки стільця має становити (300+-20) мм, ширина – не менше ніж 380 мм, радіус кривизни горизонтальної площини – 400мм. Кут нахилу спинки має регулюватися в межах 1-30 градусів від вертикального положення. Відстань від спинки до переднього краю сидіння має регулюватися в межах 260-400мм. Для зниження статичного напруження м'язів верхніх кінцівок слід використовувати стаціонарні або змінні підлокітники завдовжки не менше ніж 250мм, завширшки 50-70мм, що регулюються за висотою над сидінням у межах 230-260мм і відстанню між підлокітниками в межах 350-500мм. Поверхня сидіння і спинки стільця має бути напівм'якою з нековзним, повітронепроникним покриттям, що легко чиститься і не електризується. Робоче місце має бути обладнане підставкою для ніг завширшки не менше ніж 300мм, завглибшки не менше ніж 400мм, що регулюється за висотою в межах до 150мм і за кутом нахилу опорної поверхні підставки до 20 градусів. Підставка повинна мати рифлену поверхню і бортик по передньому краю заввишки 10мм.

Робочі місця слід розташовувати відносно світових прорізів так, щоб природне світло падало переважно з лівого боку. Монітор має розташовуватися на оптимальній відстані від очей користувача, що становить 600-700мм, але не ближче ніж за 600мм з урахуванням розміру літерно-цифрових знаків і символів. Розташування екрана монітору має забезпечувати зручність зорового спостереження у вертикальній площині під кутом +30 градусів до нормальної лінії погляду працівника. Клавіатуру слід розташовувати на поверхні столу на відстані 100-300 мм від краю, звернутого до працюючого. У конструкції клавіатури має передбачатися опорний пристрій (виготовлений із матеріалу з високим коефіцієнтом тертя, що перешкоджає мимовільному її зсуву), який дає змогу змінювати кут нахилу поверхні клавіатури у межах 5-15 градусів. Висота середнього рядка клавіш має не перевищувати 30мм. Поверхня клавіатури має бути матовою з коефіцієнтом відбиття 0,4. Розташування пристрою введення – виведення інформації має забезпечувати добру видимість монітору, зручність ручного керування в зоні досяжності моторного поля і за висотою – 900-1300мм, за шириною 400-500мм. Під матричні принтери потрібно підкладати вібраційні килимки для гасіння вібрації та шуму.

Робоче місце з персональним комп'ютером слід обладнати пюпітром для документів, що легко переміщуються.

Для забезпечення захисту і досягнення нормованих рівнів комп'ютерних випромінювань необхідно застосування приекранних фільтрів, локальних світлофільтрів

(засобів індивідуального захисту очей) та інших засобів захисту, що пройшли випробування в акредитованих лабораторіях і мають щорічний гігієнічний сертифікат.

Безпека під час роботи з персональним комп'ютером

Щодня перед початком роботи необхідно очищати монітор від пилу та інших забруднень. Після закінчення роботи персональний комп'ютер і периферійні пристрої повинні бути відключені від електричної мережі. У разі виникнення аварійної ситуації необхідно негайно відключити персональний комп'ютер і периферійні пристрої від електричної мережі.

Не допускається:

- виконувати обслуговування, ремонт та налагодження персонального комп'ютеру та периферійних пристроїв безпосередньо на робочому місці оператора;
- зберігати біля персонального комп'ютеру та периферійних пристроїв папір, будь-які носії інформації (диски, флешки тощо), запасні блоки, деталі тощо, якщо вони не використовуються для поточної роботи;
- відключати захисні пристрої, самочинно проводити зміни у конструкції та складі персонального комп'ютеру та периферійних пристроїв або їх технічне налагодження;
- працювати з персональним комп'ютером, у яких під час роботи з'являються нехарактерні сигнали, нестабільне зображення на моніторі тощо;
- працювати з матричним принтером за відсутності вібраційного килимка та зі знятою (піднятою) верхньою кришкою.

4.2 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.2.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00-7.15-18

[3] «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з ВДТ і ПП. Переважно роботи за проектами виконують у кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, зокрема персональні комп'ютери (ПК) та периферійні пристрої.

–робоча напруга U=+220В +-5%;

–робочий струм I=2А;

–споживана потужність P=350 Вт.

Робочі місця мають відповідати вимогам Державних санітарних правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин, затверджених постановою Головного державного санітарного лікаря України від 10.12.98 N 7 [8]. За умов роботи з ПК виникають наступні небезпечні та шкідливі чинники: несприятливі мікрокліматичні умови, освітлення, електромагнітні випромінювання, забруднення повітря шкідливими речовинами (джерелом, яких можуть бути: принтер, сканер та інші джерела виділення багатьох хімічних речовин - напр., озону, оксидів азоту та аерозолів високодисперсних частинок тонера), шум, вібрація, електричний струм, електростатичне поле, напруженість трудового процесу та інше.

4.2.2 Пожежна безпека

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції і кондиціювання,. Небезпека загоряння пов'язана з особливістю комп'ютерів - із значною кількістю щільно розташованих на монтажній платі і блоках електронних вузлів і схем, електричних і комутаційних кабелів, резисторів, конденсаторів, напівпровідникових діодів і транзисторів. Надійна робота окремих елементів і мікросхем в цілому забезпечується тільки в певних інтервалах температури, вологості і при заданих електричних параметрах. При відхиленні реальних умов експлуатації від розрахункових можуть виникнути пожежонебезпечні ситуації.

Запобігти утворенню горючого середовища (замінити горючі речовини і матеріали на негорючі і важко горючі) не надається технічно можливим. Тому проектом передбачаються способи і засоби запобігання утворення (або внесення) в горюче середовище джерел запалювання, таких як: застосування електроустаткування, відповідної пожежонебезпечної і вибухонебезпечної зонам відповідно до ПУЕ;

Згідно ГОСТ 12.1.004-91 ССБТ [4] таке приміщення, площею 25 м^2, відноситься до категорії "В" (пожежонебезпечної) та для протипожежного захисту в ньому проектом передбачено устаткування автоматичною пожежною сигналізацією із застосуванням датчиків-сповіщувачів РІД-1 (сповіщувач димовий ізоляційний) в кількості 1 шт., і застосуванням первинних засобів пожежогасіння.

4.2.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три- провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприй мачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.3 Гігієнічні вимоги до параметрів виробничого середовища

– площа приміщення повинна бути не менше 6,0 м² на 1 робоче місце; робочі місця повинні бути розташовані на відстані не менше ніж 1 м від стіни з вікном, і 1,4 м від звичайної стіни; відстань між бічними поверхнями комп‘ютерів має бути не меншою за 1,2 м; відстань між тильною поверхнею одного комп‘ютера та екраном іншого не повинна бути меншою 2,5м.

– відповідні робочі місця заборонено облаштовувати у підвальних або цокольних приміщеннях будинків. В обладнанні приміщень забороняється використання полімерних матеріалів (деревинно-стружкові плити, шпалери, що миються, рулонні синтетичні

матеріали, шаруватий паперовий пластик тощо), що виділяють у повітря шкідливі хімічні речовини. Покриття підлоги повинно бути матовим, а поверхня – рівною, неслизькою, з антистатичними властивостями.

– особливу увагу необхідно приділити колірній гармонії офісних приміщень. Колір є засобом створення психологічного комфорту та підвищення продуктивності праці. Найбільш сприятливі для нервової системи світлі, пастельні тони – зеленувато-блакитний, ясно-сірий, золотавий. Яскраві, контрастні поєднання (синій і жовтогарячий, червоний і фіолетовий) викликають втому, роздратування.

-у приміщеннях, де здійснюється робота з комп'ютерами, щодня має проводитися вологе прибирання з метою недопущення запиленості підлоги та меблів. Крім того, має бути обладнана кімната психологічного розвантаження.

- конструкція робочого столу та крісла користувача персонального комп'ютера має забезпечити підтримання оптимальної робочої пози та забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання (дисплея, клавіатури, принтера) і документів.

- приміщення для роботи з персональними комп'ютерами мають бути обладнані системами опалення, кондиціонування повітря, або припливно-витяжною вентиляцією. У приміщеннях на робочих місцях мають забезпечуватись оптимальні значення параметрів мікроклімату: температура повітря повинна становити 22–25°С, відносна вологість повітря — 40–60%, швидкість руху повітря — не більше 0,1 м/с. При недотриманні вказаних показників мікроклімату в офісних приміщеннях робочий день для робітників повинен бути скорочений мінімум на 10%.

- досить важливим є вимоги до освітлення приміщень,оскільки відомо, що тривала робота за комп'ютером та з документами при недостатньому рівні освітленості може призвести до значного перенапруження зору. Природне освітлення має забезпечувати коефіцієнт природної освітленості (КПО) не нижче ніж 1,5%. Для регулювання рівня освітлення природним світлом бажано застосовувати жалюзі. Робоче місце, обладнане ПК повинно бути розташоване так, щоб уникнути попадання в очі прямого сонячного світла. Штучне освітлення приміщення має бути обладнане системою загального рівномірного освітлення. Застосування світильників без розсіювачів та екрануючих сіток забороняється. Рівень освітленості на робочому столі в зоні розташування документів має бути в межах 300–500 лк.

- в офісних приміщеннях нормуються також еквівалентні рівні звуку (для програмістів – 50 дБА, а для операторів в залах обробки інформації на ПК та операторів комп'ютерного набору – 65 дБА).

-вимоги щодо рівня неіонізуючих електромагнітних випромінювань, електростатичних і магнітних полів, а також інтенсивність потоків інфрачервоного та ультрафіолетового випромінювань встановлюються відповідно до ДСанПіН 3.3.2-007-98 [1] і ДСанПіН 3.3.6.096-2002 [5].

4.4 Мікроклімат

Суттєвий вплив на стан організму працівника, його працездатність здійснює мікроклімат (метеорологічні умови) у виробничих приміщеннях, під яким розуміють умови внутрішнього середовища цих приміщень, що впливають на тепловий обмін працюючих з оточенням. Мікроклімат визначається сукупністю фізичних параметрів повітряного середовища, таких як температура, швидкість руху, вологість і барометричний тиск повітря, температура поверхонь, що оточують людину, та інтенсивність інфрачервоного випромінювання.

Властивість організму людини підтримувати тепловий баланс із навколишнім середовищем називаються терморегуляцією.

Нормальне протікання фізіологічних процесів, а отже, і хороше самопочуття можливе лише тоді, коли тепло, що виділяється організмом людини, постійно відводиться в навколишнє середовище. Кількість тепла, що утворюється в організмі людини, залежить від фізичних навантажень, а рівень тепловіддачі — від мікрокліматичних умов, переважно температури повітря.

Теплообмін організму людини з навколишнім середовищем здійснюється такими способами (шляхами): конвекція, теплопровідність, випромінювання та випаровування вологи з поверхні шкіри.

Основним нормативним документом, що регламентує параметри мікроклімату виробничих приміщень, є ДСН 3.3.6.042-99 [8]. Цей документ встановлює оптимальні і допустимі значення температури, відносної вологості та швидкості руху повітря, допустиму температуру внутрішніх поверхонь приміщення (стіни, стеля, підлога) і зовнішніх поверхонь технологічного обладнання, а також допустиму інтенсивність теплового випромінювання нагрітих поверхонь у приміщенні та відкритих джерел тепла (нагрітий метал, скло, відкритий вогонь тощо) для робочої зони — визначеного простору, в якому знаходяться робочі місця постійного або непостійного (тимчасового) перебування працівників.

Рівень шуму не перевищує санітарних норм. Тому застосування захисту від шуму в роботі не передбачається.

Ще одна проблема полягає в тому, що спектр випромінювання комп'ютерного монітора включає в себе рентгенівську, ультрафіолетову та інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівських променів мізерно мала, оскільки цей вид випромінювання поглинається речовиною екрану. Однак велику увагу слід приділяти біологічним ефектам низькочастотних електромагнітних полів.

Табл. 4.1 Оптимальні величини температури, відносної вологості та швидкості руху повітря в робочій зоні.

Період року	Категорія робіт	Температура повітря, град. С	Відносна вологість повітря, %	Швидкість руху повітря, м/с
		оптимальна	оптимальна	оптимальна
Холодний період року	легка 1 а	22 – 24	40 – 60	0.1
	легка-1 б	21 – 23	40 – 60	0.1
Теплий період року	легка-1 а	23 - 25	40 – 60	0.1
	легка-1 б	22 – 24	40 – 60	0.2

Для зниження стомлюваності персоналу в приміщеннях, де розташовані обчислювальні засоби, передбачається використовувати кольорові поєднання й покриття, що не дають відблисків.

У проекті, що розробляється передбачається використовувати сполучне освітлення. У світлий час доби приміщення буде висвітлюватися через віконні прорізи, в решту часу буде використовуватися штучне освітлення. Штучне освітлення створюється лампами розжарювання або газорозрядними лампами. Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи мають високу світловою віддачею (до 75 лм/Вт і більше), тривалим терміном служби (до 10000 годин), спектральним складом випромінюваного світла, близьким до сонячного. При експлуатації комп'ютерів виробляється зорова робота IV в розряду точності (середня точність). При цьому нормована освітленість на робочому місці дорівнює 200 лк. Джерелом природного світла (освітлення) є сонячне світло. У приміщенні, де розташовані комп'ютери, передбачається природне бічне освітлення, рівень якого відповідає ДБН В.2.5-28-2015 [6] "Будівельні норми і правила".

Регулярно проводиться контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для данного приміщення в світлий час доби достатньо природного освітлення. Для організації освітлення в темний час доби передбачається обладнати приміщення, довжина якого становить 5 м, ширина 5 м, висота 4 м, світильниками ЛП02П, облаштованими лампами типу ЛБ (дві по 80 Вт) зі світловим потоком 5400 лм кожна.

Розрахунок штучного освітлення проводиться за коефіцієнтами використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні.

Розрахунок кількості світильників здійснюється за формулою:

$$N = (E \times S \times Z \times K)/(F \times U \times M) \quad (4.1)$$

де N - число світильників;

E - нормоване освітлення;

S - площа підлоги, м^2;

Z - поправний коефіцієнт світильника (для стандартних світильників Z=1.1÷1.3);

K - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації;

U - коефіцієнт використання, що залежить від типу світильника, показника індексу приміщення і т. п. (U= 0.55 ÷ 0.6);

M - число люмінесцентних ламп у світильнику;

F - світловий потік.

Згідно вимог ДБН В.2.5-28-2015 [6], освітлення робочого місця оператора обчислювальної техніки повинно бути не менше 200 лк.

$$N = \frac{200 \times 25 \times 1{,}1 \times 1{,}5}{5500 \times 0{,}6 \times 2} = 1{,}27$$

Обираємо кількість світильників, що дорівнює 1.

4.5 Охорона навколишнього природного середовища

Діяльність за темою магістерської роботи, а саме робота за комп’ютером в процесі її виконання впливає на навколишнє природнє середовище і регламентується нормами діючого законодавства.

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на відалення (знешкодження), утилізацію, тощо в ІТ галузі.

Вплив на атмосферне повітря при нормальних умовах праці не оказує, бо не має в приміщенні сканерів, принтерів та інших джерел викиду забруднюючих речовин в повітря робочої зони.

В процесі діяльності комп’ютера виникають процеси поводження з відходами ІТ галузі. Нижче надано перелік відходів, що утворюються в процесі роботи:

- відпрацьовані люмінесцентні лампи - I клас небезпеки;
- батарейки та акумулятори (малі) -III клас небезпеки;
- змінні носії інформації - IV клас небезпеки;
- відходи друкуючих пристроїв - IV клас небезпеки;
- відпрацьований ізолюючий матеріал, дроти та кабелі - IV клас небезпеки;
- макулатура - IV клас небезпеки;
- побутові відходи - IV клас небезпеки.

Відходи в міру їх накопичення збирають у тару, відповідну класу небезпеки, з дотриманням правил безпеки, після чого доставляють до місця тимчасового зберігання відходів відповідно до затвердженої схеми їх розміщення. Зазначені для зберігання відходів місця чи об'єкти повинні використовуватися лише для заявлених відходів.

Не допускається зберігання відходів у невстановлених схемою місцях, а також перевищення норм тимчасового зберігання відходів.

Способи тимчасового зберігання відходів визначаються видом, агрегатним станом і класом небезпеки відходів:

- відходи I класу небезпеки зберігаються в герметичній тарі (сталеві бочки, контейнери). У міру наповнення тару з відходами закривають герметично сталевий кришкою;

- відходи III класу небезпеки зберігаються в тарі, яка забезпечує локалізацію зберігання, дозволяє виконувати вантажно-розвантажувальні і транспортні роботи і виключає поширення в ОС шкідливих речовин;

- відходи IV класу небезпеки можуть зберігатися відкрито на промисловому майданчику у вигляді конусоподібної купи, звідки їх автонавантажувачем перевантажують у самоскид і доставляють на місце утилізації або захоронення.

З метою визначення та прогнозування впливу відходів на навколишнє середовище, своєчасного виявлення негативних наслідків, їх запобігання відповідно до Закону України «Про відходи» [7] повинен здійснюватися моніторинг місць утворення, зберігання, і видалення відходів.

4.6 Висновки до розділу 4

У розділі «Охорона праці» проаналізовано потенційні небезпеки при роботі з засобами обчислювальної техніки, на підставі якого розроблено заходи з техніки безпеки, заходи, що забезпечують виробничу санітарію та гігієну праці, рекомендації з пожежної профілактики, які підтверджені відповідними розрахунками.

Була наведена схема, розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп'ютері.

А також визначені основні екологічні аспекти впливу на навколишнє природнє середовище та зазначені заходи щодо поводження з ними.

4.7 Перелік посилань до розділу 4

1. ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».Затверджено постановою Головного державного санітарного лік аря України 10 грудня 1998 р. № 7. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

2. НПАОП 40.1-1.01-97 «Правила безпечної експлуатації електроустановок». Наказ Державного комітету України по нагляду за охороною праці від 6 жовтня 1997 р. № 257 Зареєстровано в Міністерстві юстиції України 13 січня 1998 р. за № 11/2451. Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0011-98

3. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». Зареєстровано в Міністерстві юстиції України 25

квітня 2018 р. За № 508/31960. Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0508-18

4. ГОСТ 12.1.004-91 ССБТ «Пожежна безпека. Загальні вимоги». Затверджено і введено в дію Ухвалою Державного комітету СРСР з управління якістю продукції та станд артів від 14.06.91 №875. Режим доступу: www.URL: http://docs.cntd.ru/document/9051953

5. ДСанПіН 3.3.6.096-2002 «Державні санітарні норми і правила при роботі з джерелами електромагнітних полів». Зареєстровано в Міністерстві юстиції України 13 березня 2003 р. за № 203/7524. Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/z0203-03

6. ДБН В.2.5-28-2015 «Природнє і штучне освітлення». Режим доступу: www. URL: http://www.minregion.gov.ua/wp-content/uploads/2018/12/V2528-1.pdf

7. Закон України «Про відходи». Відомості Верховної Ради України (ВВР), 1998, № 36-37, ст.242. Режим доступу: www. URL: https://zakon.rada.gov.ua/laws/show/187/98-вр

8. ДСН 3.3.6.042-99 «Державні санітарні норми мікроклімату виробничих приміщень». Вводиться в дію Постановою Головного Державного санітарного лікаря України від 01.12.1999 № 42. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/va042282-99

ВИСНОВКИ

В даній роботі виконано дослідження підходів та методів шифрування/дешифрування з використанням еліптичних кривих, зокрема алгоритми електронно-цифрового підпису.

В межах даної роботи розглянуті еліптичні криві, параметри яких визначені над скінченними полями з простою характеристикою $GF(p)$ та бінарним розширенням $GF(2^m)$.

Виконаний огляд математичних основ побудови еліптичних кривих, опис основних алгебраїчних структур, над якими визначаються параметри точок еліптичної кривої таких як кільце, група, поле, скінченне поле та інші.

Умовно, операції, що виконуються в рамках застосування алгоритмів шифрування з використанням еліптичної криптографії можна розділити на 2 основних класи: операції нижнього та верхнього рівня, де операції нижнього рівня – це арифметичні операції над елементами скінченних полів та арифметичні операції над точками еліптичної кривої, а операції верхнього рівня, тобто ті, що використовують операції нижнього рівня – знаходження оберненого елемента та скалярне множення точки еліптичної кривої на число.

Найбільш обчислювально-витратною операцією у еліптичній криптографії є багатократне скалярне множення точки еліптичної кривої на число. Було проведено дослідження різноманітних методів багатократного скалярного множення точок еліптичної кривої на число таких як: метод SMPM (simultaneous multiple point multiplication) та його модифікації, що використовують подання множника у вигляді NAF та JSF та методів з поданням множника у вигляді DBNS. Запропоновано новий метод багатократного скалярного множення точок еліптичної кривої на число, що показує кращу швидкодію в порівнянні з існуючими методами для еліптичних кривих з параметрами, визначеними над полем $GF(2^m)$.

Розроблений програмний комплекс електронно-цифрового підпису з відкритим вихідним кодом з розширюваною програмною архітектурою, що дозволяє динамічно додавати та використовувати різні програмні реалізації методу багатократного скалярного множення точок еліптичної кривої на число. Даний комплекс має унікальну архітектуру, що задовольняє основним законам об’єктно-орієнтованого проектування SOLID.

Практична цінність отриманих в роботі результатів полягає в тому, що запропоновані методи та алгоритми дозволяють швидше виконувати операцію

багатократного скалярного множення точки еліптичної кривої над скінченним полем $GF(p)$ та $GF(2^m)$, що в свою чергу дозволяє зменшити час перевірки електронно-цифрового підпису з використанням еліптичних кривих.

Розроблене математичне та програмне забезпечення дозволяє як виконувати електронно-цифровий підпис інформаційних повідомлень довільної довжини так і проводити дослідження та порівняльний аналіз методів багатократного скалярного множення точки еліптичної кривої над скінченним полем $GF(p)$ та $GF(2^m)$.

Розроблений програмний комплекс може бути використаний в навчальному процесі кафедри комп'ютерних наук та інженерії при проведені практичних занять з дисципліни «Захист інформації в комп'ютерних системах».

ДОДАТОК А

Лістинг коду

EllipticCurveArithmetics.java

```
/*
* Copyright 2019 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.elliptic.arithmetics;
import com.trident.crypto.elliptic.EllipticCurve;
import com.trident.crypto.elliptic.EllipticCurveOperator;
import com.trident.crypto.elliptic.EllipticCurvePoint;
import com.trident.crypto.elliptic.nist.SECP;
import java.math.BigInteger;
/**
*
* defines class representing the arithmetics over provided elliptic curve, i.e.
* point addition, multiplication, doubling, belonging test
* @author trident
*/
public abstract class EllipticCurveArithmetics implements EllipticCurveOperator{
//olliptic curve over which this arithmetics is defined
protected final EllipticCurve ellipticCurve;
```

```
public EllipticCurveArithmetics(EllipticCurve ellipticCurve) {
this.ellipticCurve = ellipticCurve;
}
@Override
public EllipticCurve getEllipticCurve() {
return ellipticCurve;
}
/**
* static factory method to create the arithmetics over the elliptic curve
* defined by the standard specification
*
* should prefer this instead of constructor call
* @param spec
* @return
*/
public static EllipticCurveOperator createFrom(SECP spec){
return new PointAtInfinityArithmeticsDecorator(spec.getType()?
new ECOverPFArithmetics(EllipticCurve.createFrom(spec)):
new ECOverBEFArithmetics(EllipticCurve.createFrom(spec)));
}
/**
* provides the human readable information about this arithmetics and its elliptic curve
* @return
*/
@Override
public String toString(){
return new StringBuilder().append("Elliptic curve arithmetics defined
over:\n").append(getEllipticCurve()).toString();
}
@Override
public EllipticCurvePoint mul(BigInteger times, EllipticCurvePoint p1){
if(times.signum()==-1) throw new RuntimeException("negative times");
if(times.compareTo(BigInteger.ZERO) == 0)
throw new RuntimeException("multiply to zero");
if(times.compareTo(BigInteger.ONE) == 0)
```

```
return p1;
EllipticCurvePoint temp = EllipticCurvePoint.create(p1.getPointX(), p1.getPointY());
times = times.subtract(BigInteger.ONE);
while (times.compareTo(BigInteger.ZERO)>0){
if (times.testBit(0)){
if (temp.equals(p1))
temp = doub(temp);
else{
if(temp.equals(negate(p1))) // if adding p + (-p)
temp = EllipticCurvePoint.POINT_ON_INFINITY;
else
temp = add(temp,p1);
}
times = times.subtract(BigInteger.ONE);
}
times = times.shiftRight(1);
p1 = doub(p1);
}
return temp;
}
}
```

PointAtInfinityArithmeticsDecorator.java

```
/*
* Copyright 2018 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

```
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.elliptic.arithmetics;
import com.trident.crypto.elliptic.EllipticCurve;
import com.trident.crypto.elliptic.EllipticCurveOperator;
import com.trident.crypto.elliptic.EllipticCurvePoint;
import java.math.BigInteger;
/**
* decorator pattern usage for edge cases check while facing with
* points on infinity
* @author trident
*/
public class PointAtInfinityArithmeticsDecorator implements EllipticCurveOperator{
private final EllipticCurveArithmetics arithmetics;
public PointAtInfinityArithmeticsDecorator(EllipticCurveArithmetics arithmetics){
this.arithmetics = arithmetics;
}
@Override
public EllipticCurvePoint add(EllipticCurvePoint p1, EllipticCurvePoint p2) {
if(p1.equals(negate(p2))) return EllipticCurvePoint.POINT_ON_INFINITY;
if(p1.equals(EllipticCurvePoint.POINT_ON_INFINITY)) return p2;
if(p2.equals(EllipticCurvePoint.POINT_ON_INFINITY)) return p1;
return arithmetics.add(p1, p2);
}
@Override
public EllipticCurvePoint mul(BigInteger times, EllipticCurvePoint p1) {
if(p1.equals(EllipticCurvePoint.POINT_ON_INFINITY))
return EllipticCurvePoint.POINT_ON_INFINITY;
return arithmetics.mul(times, p1);
}
@Override
public EllipticCurvePoint doub(EllipticCurvePoint p1) {
if(p1.equals(EllipticCurvePoint.POINT_ON_INFINITY))
return EllipticCurvePoint.POINT_ON_INFINITY;
```

```
return arithmetics.doub(p1);
}
@Override
public boolean belongsTo(EllipticCurvePoint p1) {
if(p1.equals(EllipticCurvePoint.POINT_ON_INFINITY)) return true;
return arithmetics.belongsTo(p1);
}
@Override
public EllipticCurvePoint negate(EllipticCurvePoint p1) {
if(p1.equals(EllipticCurvePoint.POINT_ON_INFINITY))
return EllipticCurvePoint.POINT_ON_INFINITY;
return arithmetics.negate(p1);
}
@Override
public EllipticCurve getEllipticCurve() {
return arithmetics.getEllipticCurve();
}
@Override
public String toString(){
return arithmetics.toString();
}
}
```

ECDSA.java

```
/*
* Copyright 2018 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
```

```
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.algo;
import com.trident.crypto.algo.mpmbehavior.MPMBehavior;
import com.trident.crypto.algo.mpmbehavior.MPMStandardBehavior;
import com.trident.crypto.elliptic.EllipticCurveOperator;
import com.trident.crypto.elliptic.EllipticCurvePoint;
import java.math.BigInteger;
import java.util.Random;
/**
* implements EDSA (elliptic curve digital signature algorithm)
* @author trident
*/
public class ECDSA {
/**
* arithmetics over the elliptic curve
* @see EllipticCurveArithmetics
*/
private final EllipticCurveOperator operator;
/**
* multiple point multiplication calculation strategy
*/
private MPMBehavior mpmBehavior;
private final Random random;
public ECDSA(EllipticCurveOperator operator, MPMBehavior behavior){
this.operator = operator;
this.random = new Random();
this.mpmBehavior = behavior;
}
public ECDSA(EllipticCurveOperator operator){
this(operator, new MPMStandardBehavior(operator));
}
```

```
/**
* generate key pair i.e. secret and public
* such that
* secret = b (random BigInteger)
* public = Q = b*G (product of generator point of the elliptic curve on b)
* @return
*/
public ECDSAKey generateKeyPair(){
BigInteger sKey = new BigInteger(getOperator().getEllipticCurve().getN().bitLength(), random);
EllipticCurvePoint pKey = getOperator().mul(sKey, getOperator().getEllipticCurve().getG());
return new ECDSAKey(sKey,pKey);
}
/**
* returns the digital signature
* @param mHash - hash of the plain message M, produced using hash function such as SHA-256
* @param sKey - secret key of the key pair
* @return hexadecimal string, digital signature of mHash
*/
public String sign(byte[] mHash, BigInteger sKey){
BigInteger n = getOperator().getEllipticCurve().getN();
EllipticCurvePoint G = getOperator().getEllipticCurve().getG();
BigInteger alpha = new BigInteger(mHash);
BigInteger e = alpha.mod(n);
if(e.equals(BigInteger.ZERO)) e = BigInteger.ONE;
BigInteger k;
EllipticCurvePoint C;
BigInteger r;
BigInteger s;
do{
do {
k = new BigInteger(n.bitLength(), random);
} while (k.compareTo(BigInteger.ZERO) == -1 || k.compareTo(n) >= 1); // k<0 || k>n
```

```
C = getOperator().mul(k, G);
r = C.getPointX().mod(n);
s = r.multiply(sKey).add(k.multiply(e)).mod(n);
} while (r.equals(BigInteger.ZERO)||s.equals(BigInteger.ZERO));
return padding(r.toString(16),len16(n))+padding(s.toString(16),len16(n));
}
/**
* verify that provided signature corresponds to the provided hash
* @param mHash - hash of the plain message M, produced using hash function such as SHA-256
* @param pKey - public key of the key pair
* @param signature - hexadecimal string, digital signature of mHash
* @return if the signature is verified
*/
public boolean verify(byte[] mHash, EllipticCurvePoint pKey, String signature){
BigInteger n = getOperator().getEllipticCurve().getN();
EllipticCurvePoint G = getOperator().getEllipticCurve().getG();
BigInteger r = new BigInteger(signature.substring(0, len16(n)), 16);
BigInteger s = new BigInteger(signature.substring(len16(n), 2*len16(n)), 16);
if(r.compareTo(BigInteger.ZERO)<=0||r.compareTo(n)>=0) return false;
if(s.compareTo(BigInteger.ZERO)<=0||s.compareTo(n)>=0) return false;
BigInteger alpha = new BigInteger(mHash);
BigInteger e = alpha.mod(n);
if(e.equals(BigInteger.ZERO)) e = BigInteger.ONE;
BigInteger v = e.modInverse(n);
BigInteger z1 = (s.multiply(v)).mod(n);
BigInteger z2 = n.add(r.multiply(v).negate()).mod(n);
EllipticCurvePoint C = mpmBehavior.mpm(z1, z2, G, pKey);
BigInteger R = C.getPointX().mod(n);
return R.equals(r);
}
private EllipticCurveOperator getOperator() {
return operator;
}
private int len16(BigInteger n){
```

```
if(n.bitLength()%4==0)
return n.bitLength()/4;
else return n.bitLength()/4+1;
}
private String padding(String hexString, int byteSize){
if(byteSize<hexString.length()) throw new RuntimeException("input string length is bigger than required");
if(byteSize==hexString.length()) return hexString;
StringBuilder sb = new StringBuilder(hexString).reverse();
while (sb.length()<byteSize) {
sb.append("0");
}
return sb.reverse().toString();
}
public MPMBehavior getMpmBehavior() {
return mpmBehavior;
}
public void setMpmBehavior(MPMBehavior mpmBehavior) {
if(mpmBehavior == null) throw new RuntimeException("should not be null");
this.mpmBehavior = mpmBehavior;
}
}
```

EllipticCurve.java

```
/*
* Copyright 2018 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
```

```
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.elliptic;
import com.trident.crypto.elliptic.nist.SECP;
import com.trident.crypto.field.element.BinaryExtensionFieldElement;
import com.trident.crypto.field.element.BinaryExtensionFieldElementFactory;
import com.trident.crypto.field.element.FiniteFieldElement;
import com.trident.crypto.field.element.FiniteFieldElementFactory;
import com.trident.crypto.field.operator.FiniteFieldElementArithmetics;
import java.math.BigInteger;
/**
* @see http://www.secg.org/SEC2-Ver-1.0.pdf
* @author trident
*/
public class EllipticCurve{
/**
* parameter a of the curve equation
*/
private final FiniteFieldElement a;
/**
* parameter b of the curve equation
*/
private final FiniteFieldElement b;
/**
* point on the curve with high order
*/
private final EllipticCurvePoint G;
/**
* order of the point G
*/
private final BigInteger n;
/**
```

```
* cofactor - relation between number of points of curve and order of point G
*/
private final BigInteger h;
/**
* arithmetics of the finite field over which this curve is defined
* i.e. GF(p) or GF(2^m)
*/
private final FiniteFieldElementArithmetics fieldArithmetics;
public EllipticCurve(FiniteFieldElementArithmetics fieldArithmetics, FiniteFieldElement
a, FiniteFieldElement b,
EllipticCurvePoint G, BigInteger n, BigInteger h) {
this.a = a;
this.b = b;
this.G = G;
this.n = n;
this.h = h;
this.fieldArithmetics = fieldArithmetics;
}
public FiniteFieldElementArithmetics getFieldArithmetics() {
return fieldArithmetics;
}
public FiniteFieldElement getA() {
return a;
}
public FiniteFieldElement getB() {
return b;
}
public EllipticCurvePoint getG() {
return G;
}
public BigInteger getN() {
return n;
}
public BigInteger getH() {
return h;
```

```
}
@Override
public String toString(){
StringBuilder sb = new StringBuilder();
sb.append("Elliptic curve with:")
.append("\n")
.append(getFieldArithmetics())
.append("\n")
.append("With params:\n")
.append("A = ")
.append(getA())
.append("\n")
.append("B = ")
.append(getB())
.append("\n")
.append("G = ")
.append(getG())
.append("\n")
.append("n = ")
.append(getN())
.append("\n")
.append("H = ")
.append(getH())
.append("\n");
return sb.toString();
}
/**
* static factory method producing elliptic curve from
* standard specification
*
* should prefer this over custom constructor call
* @param spec
* @return
*/
public static EllipticCurve createFrom(SECP spec){
```

```
return createFrom(spec, spec.getType()?new FiniteFieldElementFactory():new
BinaryExtensionFieldElementFactory());
}
private static EllipticCurve createFrom(SECP spec, FiniteFieldElementFactory factory){
FiniteFieldElementArithmetics arithmetics =
FiniteFieldElementArithmetics.createFieldElementArithmetics(spec.getType()?new
BigInteger(spec.getP(),16):BinaryExtensionFieldElement.fromString(spec.getP()));
return new EllipticCurve(arithmetics,
factory.createFrom(spec.getA()),
factory.createFrom(spec.getB()),
EllipticCurvePoint.create(factory.createFrom(spec.getGx()),
factory.createFrom(spec.getGy())),
spec.getN(),
spec.getH());
}
}
```

SECP.java

```
/*
* Copyright 2018 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.elliptic.nist;
import java.math.BigInteger;
```

```
/**
* Recommended Elliptic Curve parameters by SECP
* @see http://www.secg.org/sec2-v2.pdf
* @author trident
*/
public enum SECP {
SECP112R1(
"DB7C2ABF 62E35E66 8076BEAD 208B",
"DB7C2ABF 62E35E66 8076BEAD 2088",
"659EF8BA 043916EE DE891170 2B22",
"09487239 995A5EE7 6B55F9C2 F098",
"A89CE5AF 8724C0A2 3E0E0FF7 7500",
"DB7C2ABF 62E35E76 28DFAC65 61C5",
"01",true
),
SECP112R2(
"DB7C2ABF 62E35E66 8076BEAD 208B",
"6127C24C 05F38A0A AAF65C0E F02C",
"51DEF181 5DB5ED74 FCC34C85 D709",
"4BA30AB5 E892B4E1 649DD092 8643",
"ADCD46F5 882E3747 DEF36E95 6E97",
"36DF0AAF D8B8D759 7CA10520 D04B",
"04",true
),
SECP128R1(
"FFFFFFFD FFFFFFFF FFFFFFFF FFFFFFFF",
"FFFFFFFD FFFFFFFF FFFFFFFF FFFFFFFC",
"E87579C1 1079F43D D824993C 2CEE5ED3",
"161FF752 8B899B2D 0C28607C A52C5B86",
"CF5AC839 5BAFEB13 C02DA292 DDED7A83",
"FFFFFFFE 00000000 75A30D1B 9038A115",
"01",true
),
SECP128R2(
"FFFFFFFD FFFFFFFF FFFFFFFF FFFFFFFF",
```

```
        "D6031998 D1B3BBFE BF59CC9B BFF9AEE1",
        "5EEEFCA3 80D02919 DC2C6558 BB6D8A5D",
        "7B6AA5D8 5E572983 E6FB32A7 CDEBC140",
        "27B6916A 894D3AEE 7106FE80 5FC34B44",
        "3FFFFFFF 7FFFFFFF BE002472 0613B5A3",
        "04",true),
        SECP192K1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFEE37",
        "00000000 00000000 00000000 00000000 00000000 00000000",
        "00000000 00000000 00000000 00000000 00000000 00000003",
        "DB4FF10E C057E9AE 26B07D02 80B7F434 1DA5D1B1 EAE06C7D",
        "9B2F2F6D 9C5628A7 844163D0 15BE8634 4082AA88 D95E2F9D",
        "FFFFFFFF FFFFFFFF FFFFFFFE 26F2FC17 0F69466A 74DEFD8D",
        "01",true),
        SECP192R1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF",
        "FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFC",
        "64210519 E59C80E7 0FA7E9AB 72243049 FEB8DEEC C146B9B1",
        "188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012",
        "07192B95 FFC8DA78 631011ED 6B24CDD5 73F977A1 1E794811",
        "FFFFFFFF FFFFFFFF FFFFFFFF 99DEF836 146BC9B1 B4D22831",
        "01",true),
        SECP224K1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE
FFFFE56D", "00000000 00000000 00000000 00000000 00000000 00000000 00000000",
        "00000000 00000000 00000000 00000000 00000000 00000000 00000005",
        "A1455B33 4DF099DF 30FC28A1 69A467E9 E47075A9 0F7E650E B6B7A45C",
        "7E089FED 7FBA3442 82CAFBD6 F7E319F7 C0B0BD59 E2CA4BDB 556D61A5",
        "01 00000000 00000000 00000000 0001DCE8 D2EC6184 CAF0A971 769FB1F7",
        "01",true),
        SECP224R1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 00000000
00000001", "FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFE",
        "B4050A85 0C04B3AB F5413256 5044B0B7 D7BFD8BA 270B3943 2355FFB4",
        "B70E0CBD 6BB4BF7F 321390B9 4A03C1D3 56C21122 343280D6 115C1D21",
        "BD376388 B5F723FB 4C22DFE6 CD4375A0 5A074764 44D58199 85007E34",
        "FFFFFFFF FFFFFFFF FFFFFFFF FFFF16A2 E0B8F03E 13DD2945 5C5C2A3D",
        "01",true),
```

SECP256K1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F", "00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000", "00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000007", "79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798", "483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8", "FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141", "01",true),

SECP256R1("FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFF", "FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFC", "5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D2604B", "6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296", "4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51F5", "FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3B9CAC2 FC632551", "01",true),

SECP384R1("FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFF", "FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF 00000000 00000000 FFFFFFFC", "B3312FA7 E23EE7E4 988E056B E3F82D19 181D9C6E FE814112 0314088F 5013875A C656398D 8A2ED19D 2A85C8ED D3EC2AEF", "AA87CA22 BE8B0537 8EB1C71E F320AD74 6E1D3B62 8BA79B98 59F741E0 82542A38 5502F25D BF55296C 3A545E38 72760AB7", "3617DE4A 96262C6F 5D9E98BF 9292DC29 F8F41DBD 289A147C E9DA3113 B5F0B8C0 0A60B1CE 1D7E819D 7A431D7C 90EA0E5F", "FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF C7634D81 F4372DDF 581A0DB2 48B0A77A ECEC196A CCC52973", "01",true),

SECP521R1("01FF FFFFFFFC", "0051 953EB961 8E1C9A1F 929A21A0 B68540EE A2DA725B 99B315F3 B8B48991 8EF109E1 56193951 EC7E937B 1652C0BD 3BB1BF07 3573DF88 3D2C34F1 EF451FD4 6B503F00", "00C6 858E06B7 0404E9CD 9E3ECB66 2395B442 9C648139 053FB521 F828AF60 6B4D3DBA A14B5E77 EFE75928 FE1DC127 A2FFA8DE 3348B3C1 856A429B F97E7E31 C2E5BD66", "0118 39296A78 9A3BC004 5C8A5FB4 2C7D1BD9 98F54449 579B4468 17AFBD17 273E662C 97EE7299 5EF42640 C550B901 3FAD0761 353C7086 A272C240 88BE9476 9FD16650", "01FF FFFFFFFF

FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFA 51868783 BF2F966B 7FCC0148 F709A5D0 3BB5C9B8 899C47AE BB6FB71E 91386409", "01",true),

SECT113R1("x^113 x^9 1", "00308825 0CA6E7C7 FE649CE8 5820F7",
"00E8BEE4 D3E22607 44188BE0 E9C723",
"009D7361 6F35F4AB 1407D735 62C10F",
"00A52830 277958EE 84D1315E D31886",
"01000000 00000000 D9CCEC8A 39E56F",
"02",false),

SECT163K1("x^163 x^7 x^6 x^3 1", "00 00000000 00000000 00000000 00000000 00000001", "00 00000000 00000000 00000000 00000000 00000001",
"02 FE13C053 7BBC11AC AA07D793 DE4E6D5E 5C94EEE8",
"02 89070FB0 5D38FF58 321F2E80 0536D538 CCDAA3D9",
"04 00000000 00000000 00020108 A2E0CC0D 99F8A5EF",
"02",false),

SECT163R1("x^163 x^7 x^6 x^3 1",
"07 B6882CAA EFA84F95 54FF8428 BD88E246 D2782AE2",
"07 13612DCD DCB40AAB 946BDA29 CA91F73A F958AFD9",
"03 69979697 AB438977 89566789 567F787A 7876A654",
"00 435EDB42 EFAFB298 9D51FEFC E3C80988 F41FF883",
"03 FFFFFFFF FFFFFFFF FFFF48AA B689C29C A710279B",
"02",false),

SECT163R2("x^163 x^7 x^6 x^3 1",
"00 00000000 00000000 00000000 00000000 00000001",
"02 0A601907 B8C953CA 1481EB10 512F7874 4A3205FD",
"03 F0EBA162 86A2D57E A0991168 D4994637 E8343E36",
"00 D51FBC6C 71A0094F A2CDD545 B11C5C0C 797324F1",
"04 00000000 00000000 000292FE 77E70C12 A4234C33",
"02",false),

SECT233K1("x^233 x^74 1",
"0000 00000000 00000000 00000000 00000000 00000000 00000000 00000000",
"0000 00000000 00000000 00000000 00000000 00000000 00000000 00000001",
"0172 32BA853A 7E731AF1 29F22FF4 149563A4 19C26BF5 0A4C9D6E EFAD6126",
"01DB 537DECE8 19B7F70F 555A67C4 27A8CD9B F18AEB9B 56E0C110 56FAE6A3", "80 00000000 00000000 00000000 00069D5B B915BCD4 6EFB1AD5 F173ABDF", "04",false),

```
SECT233R1("x^233 x^74 1",
"0000 00000000 00000000 00000000 00000000 00000000 00000001",
"0066 647EDE6C 332C7F8C 0923BB58 213B333B 20E9CE42 81FE115F 7D8F90AD",
"00FA C9DFCBAC 8313BB21 39F1BB75 5FEF65BC 391F8B36 F8F8EB73 71FD558B",
"0100 6A08A419 03350678 E58528BE BF8A0BEF F867A7CA 36716F7E 01F81052",
"0100 00000000 00000000 00000000 0013E974 E72F8A69 22031D26 03CFE0D7",
"02",false),
SECT239K1("x^239 x^158 1",
"0000 00000000 00000000 00000000 00000000 00000000 00000000",
"0000 00000000 00000000 00000000 00000000 00000000 00000001",
"29A0 B6A887A9 83E97309 88A68727 A8B2D126 C44CC2CC 7B2A6555 193035DC",
"7631 0804F12E 549BDB01 1C103089 E73510AC B275FC31 2A5DC6B7 6553F0CA",
"2000 00000000 00000000 00000000 005A79FE C67CB6E9 1F1C1DA8 00E478A5",
"04",false),
SECT283K1("x^283 x^12 x^7 x^5 1",
"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000",
"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001",
"0503213F 78CA4488 3F1A3B81 62F188E5 53CD265F 23C1567A 16876913 B0C2AC24 58492836",
"01CCDA38 0F1C9E31 8D90F95D 07E5426F E87E45C0 E8184698 E4596236 4E341161 77DD2259",
"01FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFE9AE 2ED07577 265DFF7F 94451E06 1E163C61", "04",false),
SECT283R1("x^283 x^12 x^7 x^5 1",
"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001",
"027B680A C8B8596D A5A4AF8A 19A0303F CA97FD76 45309FA2 A581485A F6263E31 3B79A2F5",
"05F93925 8DB7DD90 E1934F8C 70B0DFEC 2EED25B8 557EAC9C 80E2E198 F8CDBECD 86B12053",
"03676854 FE24141C B98FE6D4 B20D02B4 516FF702 350EDDB0 826779C8 13F0DF45 BE8112F4",
```

"03FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFEF90 399660FC 938A9016 5B042A7C EFADB307", "02",false),

SECT409K1("x^409 x^87 1",

"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000 00000000",

"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000 00000001",

"0060F05F 658F49C1 AD3AB189 0F718421 0EFD0987 E307C84C 27ACCFB8 F9F67CC2 C460189E

B5AAAA62 EE222EB1 B35540CF E9023746",

"01E36905 0B7C4E42 ACBA1DAC BF04299C 3460782F 918EA427 E6325165 E9EA10E3 DA5F6C42

E9C55215 AA9CA27A 5863EC48 D8E0286B",

"7FFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFE5F 83B2D4EA 20400EC4

557D5ED3 E3E7CA5B 4B5C83B8 E01E5FCF", "04",false),

SECT409R1("x^409 x^87 1",

"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

00000000 00000000 00000001",

"0021A5C2 C8EE9FEB 5C4B9A75 3B7B476B 7FD6422E F1F3DD67 4761FA99 D6AC27C8 A9A197B2

72822F6C D57A55AA 4F50AE31 7B13545F",

"015D4860 D088DDB3 496B0C60 64756260 441CDE4A F1771D4D B01FFE5B 34E59703 DC255A86

8A118051 5603AEAB 60794E54 BB7996A7",

"0061B1CF AB6BE5F3 2BBFA783 24ED106A 7636B9C5 A7BD198D 0158AA4F 5488D08F 38514F1F

DF4B4F40 D2181B36 81C364BA 0273C706",

"01000000 00000000 00000000 00000000 00000000 00000000 000001E2 AAD6A612 F33307BE 5FA47C3C

9E052F83 8164CD37 D9A21173",

"02",false),

```
SECT571K1(
"x^571 x^10 x^5 x^2 1",
"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000",
"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000001",
"026EB7A8 59923FBC 82189631 F8103FE4 AC9CA297 0012D5D4 60248048 01841CA4 43709584
93B205E6 47DA304D B4CEB08C BBD1BA39 494776FB 988B4717 4DCA88C7 E2945283 A01C8972",
"0349DC80 7F4FBF37 4F4AEADE 3BCA9531 4DD58CEC 9F307A54 FFC61EFC 006D8A2C 9D4979C0
AC44AEA7 4FBEBBB9 F772AEDC B620B01A 7BA7AF1B 320430C8 591984F6 01CD4C14 3EF1C7A3",
"02000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 131850E1
F19A63E4 B391A8DB 917F4138 B630D84B E5D63938 1E91DEB4 5CFE778F 637C1001",
"04",false
);
private final String p;
private final BigInteger a;
private final BigInteger b;
private final BigInteger Gx;
private final BigInteger Gy;
private final BigInteger n;
private final BigInteger h;
private final boolean type;
private SECP(String p, String a, String b, String Gx, String Gy, String n, String h, boolean type) {
this.p = type?p.replace(" ", ""):p;
this.a = new BigInteger(a.replace(" ", ""), 16);
this.b = new BigInteger(b.replace(" ", ""), 16);
```

```
this.Gx = new BigInteger(Gx.replace(" ", ""), 16);
this.Gy = new BigInteger(Gy.replace(" ", ""), 16);
this.n = new BigInteger(n.replace(" ", ""), 16);
this.h = new BigInteger(h.replace(" ", ""), 16);
this.type = type;
}
/**
* @return type of these elliptic curve parameters
* true if defined over prime field
* false if defined over binary extension field
*/
public boolean getType() {
return type;
}
/**
* @return
* if prime field - > order of the field
* if binary extension field - > irreducible polynomial in x^n notation
*
*/
public String getP() {
return p;
}
/**
*
* @return parameter a of elliptic curve equation
*/
public BigInteger getA() {
return a;
}
/**
*
* @return parameter b of elliptic curve equation
*/
public BigInteger getB() {
```

```
return b;
}
/**
*
* @return x coordinate of generator point
*/
public BigInteger getGx() {
return Gx;
}
/**
*
* @return y coordinate of generator point
*/
public BigInteger getGy() {
return Gy;
}
/**
*
* @return order of generator point
*/
public BigInteger getN() {
return n;
}
/**
*
* @return cofactor i.e. h = order of field/n
*/
public BigInteger getH() {
return h;
}
}
```

FiniteFieldElementArithmetics.java

```
/*
```

```
* Copyright 2019 trident.
*
* Licensed under the Apache License, Version 2.0 (the "License");
* you may not use this file except in compliance with the License.
* You may obtain a copy of the License at
*
* http://www.apache.org/licenses/LICENSE-2.0
*
* Unless required by applicable law or agreed to in writing, software
* distributed under the License is distributed on an "AS IS" BASIS,
* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
* See the License for the specific language governing permissions and
* limitations under the License.
*/
package com.trident.crypto.field.operator;
import com.trident.crypto.field.BinaryExtensionField;
import com.trident.crypto.field.FiniteField;
import com.trident.crypto.field.PrimeField;
import com.trident.crypto.field.element.BinaryExtensionFieldElement;
import com.trident.crypto.field.element.BinaryExtensionFieldElementFactory;
import com.trident.crypto.field.element.FiniteFieldElement;
import com.trident.crypto.field.element.FiniteFieldElementFactory;
import java.math.BigInteger;
/**
* arithmetics of elements in prime field
* @author trident
*/
public abstract class FiniteFieldElementArithmetics{
private final FiniteField field;
private final FiniteFieldElementFactory elementFactory;
FiniteFieldElementArithmetics(FiniteField field, FiniteFieldElementFactory elementFactory){
this.field = field;
this.elementFactory = elementFactory;
```

```
}
/**
* add two finite field elements
* @param el1
* @param el2
* @return sum of elements
*/
public abstract FiniteFieldElement add(FiniteFieldElement el1, FiniteFieldElement el2);
/**
* subtract two finite field elements
* @param el1
* @param el2
* @return
*/
public abstract FiniteFieldElement sub(FiniteFieldElement el1, FiniteFieldElement el2);
/**
* multiply finite field elements
* @param el1
* @param el2
* @return multiple of elements
*/
public abstract FiniteFieldElement mul(FiniteFieldElement el1, FiniteFieldElement el2);
/**
* find inverse of element
* @param el1
* @return inverse
*/
public abstract FiniteFieldElement inv(FiniteFieldElement el1);
/**
* find the rest of element which belongs to the field
* @param el1
* @return
*/
public abstract FiniteFieldElement mod(FiniteFieldElement el1);
/**
```

```
        * return the element x such that el1 + x = 0 mod order
        * @param el1
        * @return
        */
        public abstract FiniteFieldElement complement(FiniteFieldElement el1);
        /**
        *
        * @return the field over which this arithmetics is performed
        */
        public FiniteField getField(){
        return field;
        }
        /**
        *
        * @return the factory producing the elements of this field
        */
        public FiniteFieldElementFactory getElementFactory(){
        return elementFactory;
        }
        @Override
        public String toString(){
        return "Arithmetics defined over field:"+getField();
        }
        /**
        * static factory method to create the arithmetics based on order
        * if fieldOrder instanceof BigInteger -> creates PrimeFieldElementArithmetics
        * if fieldOrder instanceof BinaryExtensionFieldElement -> creates
BinaryExtensionFieldElementArithmetics
        * @param fieldOrder
        * @return
        */
        public static FiniteFieldElementArithmetics createFieldElementArithmetics(BigInteger
fieldOrder){
        if(fieldOrder instanceof BinaryExtensionFieldElement) return
        createFieldElementArithmetics((BinaryExtensionFieldElement)fieldOrder);
```

```
        return new PrimeFieldElementArithmetics(new PrimeField(fieldOrder), new
FiniteFieldElementFactory());
        }
        public static FiniteFieldElementArithmetics
createFieldElementArithmetics(BinaryExtensionFieldElement
        fieldIrreduciblePoly){
        return new BinaryExtensionFieldElementArithmetics(new
BinaryExtensionField(fieldIrreduciblePoly), new
        BinaryExtensionFieldElementFactory());
```

ДОДАТОК Б

ПРЕЗЕНТАЦІЯ

Міністерство освіти і науки України
Східноукраїнський національний університет імені Володимира Даля
Факультет інформаційних технологій та електроніки
кафедра комп'ютерних наук та інженерії

«ДОСЛІДЖЕННЯ АЛГОРИТМУ СТВОРЕННЯ ЕЛЕКТРОННОГО ЦИФРОВОГО ПІДПИСУ З ВИКОРИСТАННЯМ ГРУПИ ТОЧОК ЕЛІПТИЧНОЇ КРИВОЇ»

Студент гр. КН-18 дм **Сандулов В. Ю.**
Керівник проекту **доц., к.т.н. Кардашук В.С.**

- Актуальність створення та дослідження новітніх методів забезпечення безпеки електронного документообігу, питання захисту та забезпечення достовірності даних та документів, що передаються незахищеними каналами зв'язку, такими як мережа Інтернет, займає чільне місце серед досліджень, що проводяться в даній галузі.
- **Метою роботи** є розроблення та дослідження методу високошвидкісного множення точок еліптичної кривої на число.
- **Об'єкт дослідження** – процеси обміну ключами, шифрування, дешифрування та створення/перевірка електронного цифрового підпису (ЕЦП) в криптосистемах на основі еліптичних кривих.
- **Предмет дослідження** – методи багатократного скалярного множення точок еліптичної кривої у полі *GF(p)* та $GF(2^m)$.
- **Методи дослідження** – методи теорії чисел, аналітичної геометрії, абстрактної алгебри, дискретної математики та криптографії.

Основні задачі магістерської роботи

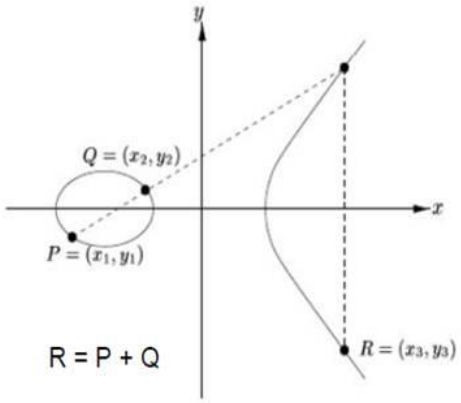
- З метою реалізації задачі дослідження:
- Провести аналіз сучасних підходів до формування криптосистем на базі еліптичних кривих та методів розподілу ключів в асиметричних криптосистемах.
-
- Визначити вплив багатократного скалярного множення точок еліптичної кривої на число на швидкість створення ЕЦП.
- Провести дослідження операцій над точками еліптичних кривих в скінченних полях та особливостей програмної реалізації;
- З метою дослідження розробити програмний комплекс для реалізації операцій над точками еліптичних кривих.
- Оцінити наукову цінність отриманих результатів та перспективи їх впровадження для практичного використання та у навчальний процес.

Теоретичні відомості

Еліптичною кривою називається крива, визначена наступним рівнянням:

$y^2 = x^3 + ax + b$ для полів з простою характеристикою GF(p)

$y^2 + xy = x^3 + ax^2 + b, \quad b \neq 0$ для полів виду GF(2^m)



R = k•Q = Q+Q+Q…(k разів)

Q+(-Q) = O (точка на нескінченності)

АНАЛІЗ ОПЕРАЦІЙ НАД ТОЧКАМИ ЕЛІПТИЧНИХ КРИВИХ

В межах криптографії, активне застосування знаходять еліптичні криві з параметрами, визначеними над скінченними полями, а саме полями з простою характеристикою – GF(p) та бінарним розширенням – $GF(2^m)$. Це в свою чергу означає, що координати точок еліптичної кривої належать деякому скінченному полю, що знімає проблему округлення значень при виконанні операцій над ними

Обчислюється точка C=k•G де k - велике випадкове число, G - точка еліптичної кривої великого порядку.
Пара C,k називається публічним та приватним ключами

Знаходження k при відомому C (ECDLP)
має велику обчислювальну складність: $O(\exp(c(\log p \log\log p)^d))$

Використання: алгоритм електронно-цифрового підпису (ECDSA)

Під час виконання алгоритму необхідно виконати обчислення $R = k•P + l•Q$, що є найбільш ресурсоємкою операцією

Обчислення $R = k•P + l•Q$ називається багатократним скалярним множенням точки еліптичної кривої на число, англ. multiple point multiplication

Існуючі підходи

1. Метод SMPM – одночасне множинне множення точок (simultaneous multiple point multiplication).

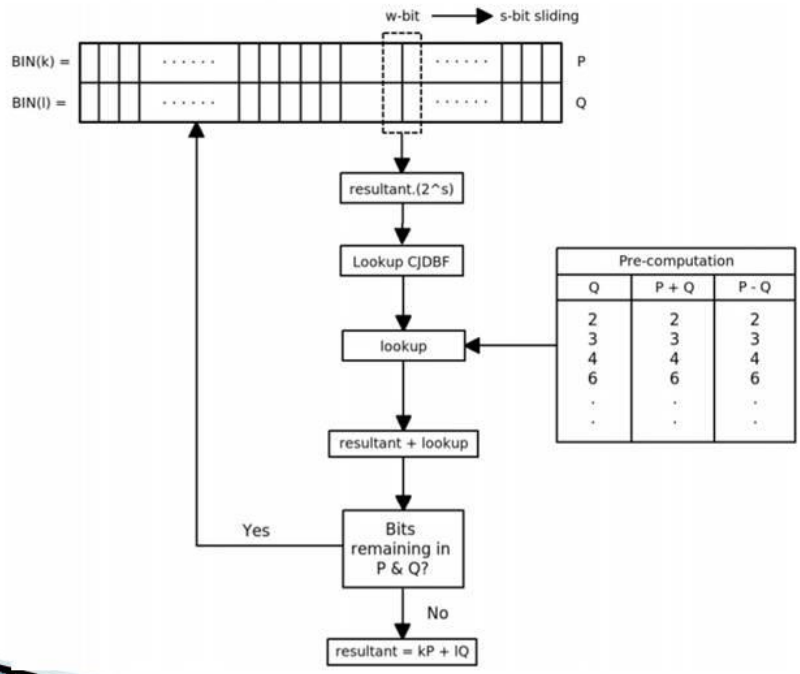
2. Метод, що базується на використанні форми DBNS.
Двобазисна система подання числа (англ. DBNS– double based number system) – це схема подання чисел, при якій кожне невід'ємне ціле число n представляється у вигляді сум та різниць добутків степенів 2 та 3.
Математично дане представлення можна зобразити як:

$$n = \sum_{i=1}^{M} s_i 2^{b_i} 3^{t_i}$$

3. Метод, що базується на використанні форми JSF(**Joint Sparse Form**).
Спільна розріджена форма. Метод заснований на одночасному перекодуванні набору цілих чисел, щоб мінімізувати кількість загальних множень чи точкових додавань (виграш до 18%).

4. Метод, що базується на використанні NAF (non-adjacent form). Насуміжні форми числа. NAF гарантує унікальне подання цілого числа, але головна перевага цього полягає в тому, що вага значення Хеммінга буде мінімальним.

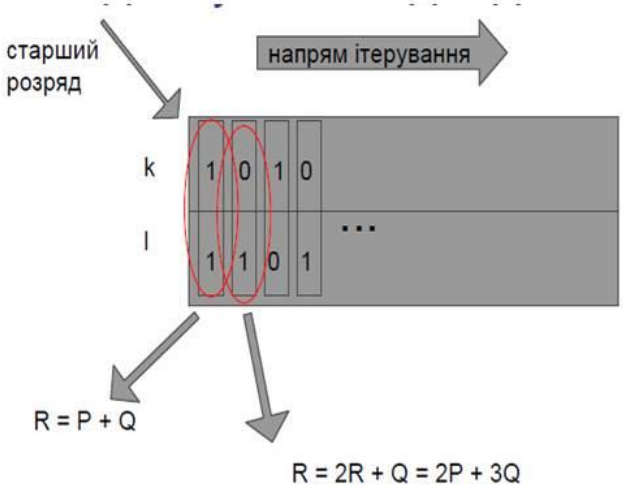
Схема роботи методу, що базується на використанні форми DBNS



NAF (non-adjacent form) це знаково-розрядна форма подання числа, тобто подання вигляду:

$$\sum_{i=0}^{l} x_i d_i^{\,i},\ x_i \in \{1, -1, 0\}$$

JSF (joint sparse form) це покращення NAF, що дозволяє ще більше зменшити кількість ненульових розрядів.



В методі SMPM виконується аналогічна операція, але не над розрядом, а над блоком (w-біт), тому необхідне передобчислення.

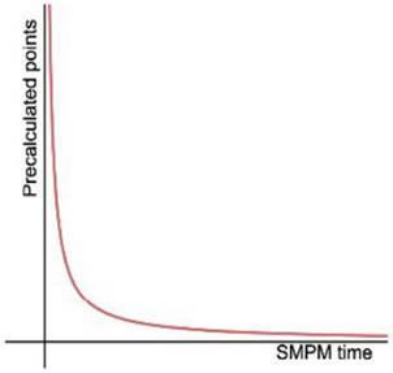
Підсумки аналізу та пропозиції

Загалом, для методів з використанням NAF та JSF необхідне передобчислення всього 4-х значень:

$$\{P+Q;\ P-Q;\ -P-Q;\ -P+Q\}$$

З іншого боку, кількість передобчислень при використанні SMPM пропорційна розміру блоку.

Хоч швидкодія SMPM і вища у порівнянні з JSF та NAF, кількість передобчислень може займати нераціонально великий час.



Для прискорення виконання методу багатократного скалярного множення пропонується передобчислення, що дозволяє уникнути операції зсуву результату вліво, тобто множення на 2.

Для цього пропонується виконати передобчислення:

$$2^n(d_iP + d_jQ),\ d_{ij} \in \{-1, 0, 1\}.\quad n \in [0, l)$$

Таким чином, кількість передобчислень зростає до 8n, але загальна швидкодія є значно кращою.

Запропонований метод названо JSFImp (JSF Improved)

Вибір параметрів еліптичної кривої

NIST (Національний інститут стандартів і технології — національний орган зі стандартизації у США) склав список еліптичних кривих з уже відомою кількістю точок, які рекомендовано використовувати в схемах ЕЦП.

Для опису кривої в стандарті NIST використовується набір з 6 параметрів

D = (p, a, b, G, n, h),

де: p – просте число, модуль еліптичної кривої;
a, b – задають рівняння еліптичної кривої;
G – точка еліптичної кривої великого порядку;
n – порядок точки G;
h – параметр, так званий кофактор. Визначається відношенням загального числа точок на еліптичній кривій до порядку точки G.

Bit length of n	Maximum Cofactor (h)
160 - 223	2^{10}
224 - 255	2^{14}
256 - 383	2^{16}
384 - 511	2^{24}
≥ 512	2^{32}

Порівняльний аналіз

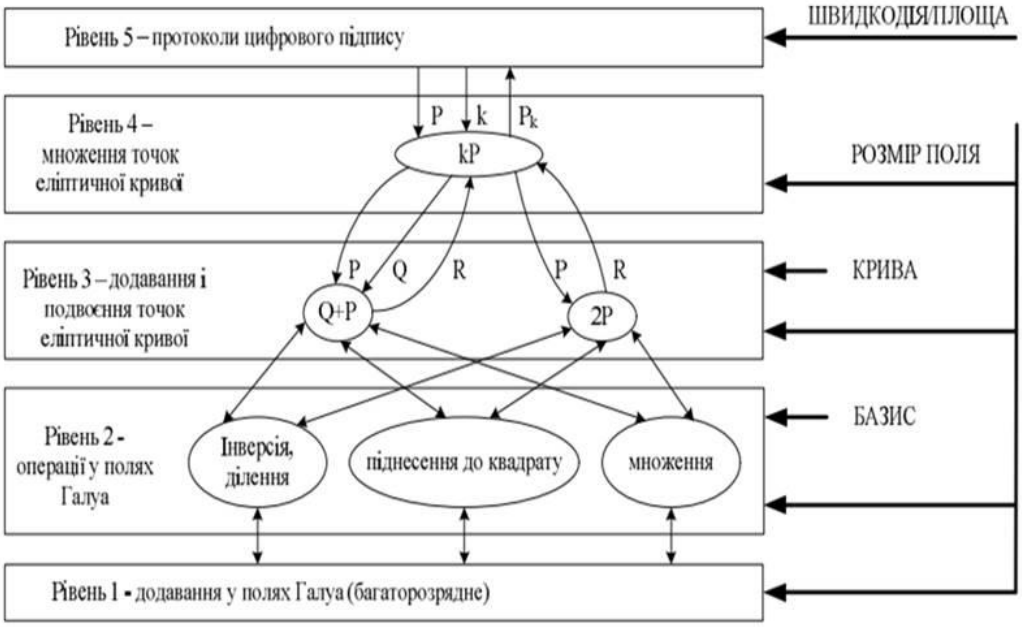
Значення часу перевірки електронно-цифрового підпису для еліптичних кривих з параметрами визначеними над полем GF(p), (мс)

	SMPM4	SMPM5	SMPM6	SMPM7	NONE	JSF	NAF	JSFImp
SECP112R1	11,9	10,6	7,2	7,9	9,2	16,7	12,8	4,2
SECP112R2	4,3	3,9	6,7	5,1	7,3	16,4	16,3	3,2
SECP128R1	11,1	5,5	6,2	8,5	7,9	10,4	25,3	5
SECP128R2	5,3	5,2	4,8	25,4	13,4	9,5	11,6	4,3
SECP192K1	19,3	13,9	18,3	12,5	14,7	26,4	29,4	11,4
SECP192R1	13,8	14,6	17,1	20,5	16	22,5	22,1	11,6
SECP224K1	31,3	16,4	15,5	13,8	19,8	30,9	28,8	16,6
SECP224R1	21,2	13,9	14	14,8	25,3	30,1	32,3	17,9
SECP256K1	19,2	25,8	18,5	19,4	25,8	42,8	46	25
SECP256R1	18,8	18,4	19,5	23,3	29,7	54,6	40,7	23,5
SECP384R1	49,9	50,8	46,4	47	68,3	108,2	114,9	66,8
SECP521R1	110,6	123,1	107,9	106,1	145	246,3	238,7	166

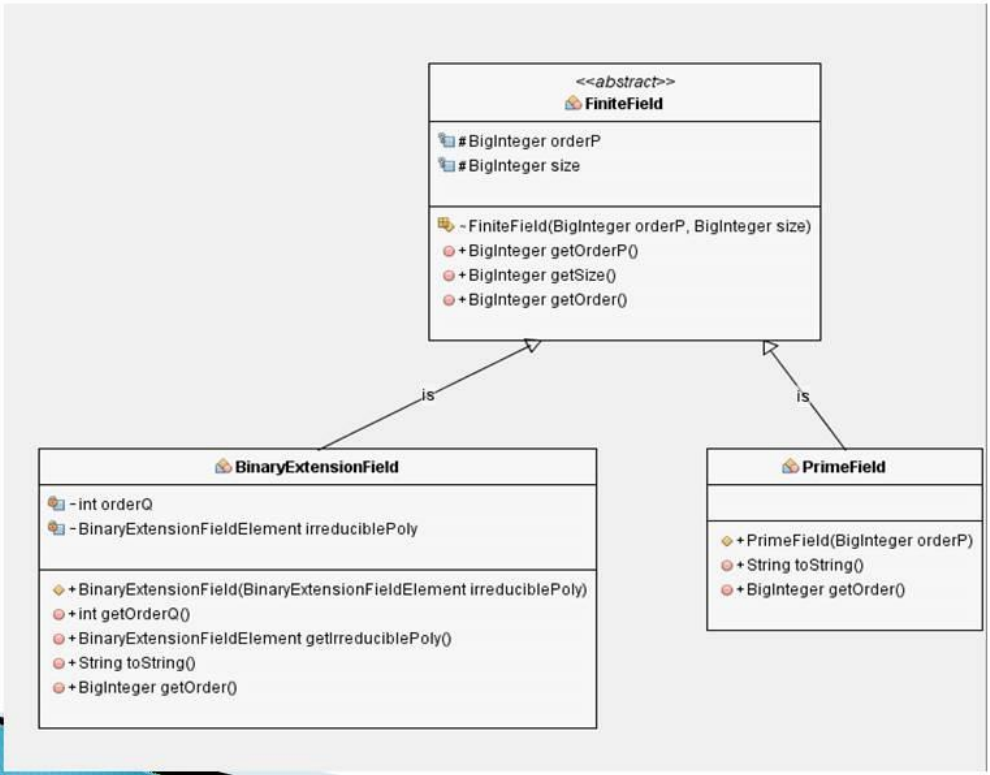
Значення часу перевірки електронно-цифрового підпису для еліптичних кривих з параметрами визначеними над полем $GF(2^m)$, (мс)

	SMPM 4	SMPM 5	SMPM 6	SMPM7	NONE	JSF	NAF	JSFImp
SECT113R1	119,63	132,76	113,79	107,37	394,04	272,85	212,08	26,72
SECT163K1	382,31	448,27	343,17	308,83	417,16	437,67	398,19	71,92
SECT163R1	348,89	284,97	329,49	310,63	385,1	378,53	349,87	82,76
SECT163R2	349,55	284,17	296,77	336,07	399,61	333,97	349,3	77,83
SECT233K1	882,59	1027,2	683,72	1066,35	922,84	829,23	799,91	164,93
SECT233R1	1066,4	1052,14	969,35	818,54	1241,4	901,1	880,04	172,2
SECT239K1	791,88	807,24	834,56	751,31	1255,24	1061,03	1089,73	191,7
SECT283K1	1141,25	1133,76	1166,88	1132,89	1852,08	1326,8	1339,85	302,2
SECT283R1	1179,77	1123,88	1136,88	1216,29	1558,77	1470,38	1479,51	296,15
SECT409K1	3187,36	3244,52	3235,17	3044,42	4202,8	3540,57	4149,11	797,64
SECT409R1	3022,08	3307,61	2810,1	3047,54	3900,77	3670,78	3452,5	856,4
SECT571K1	7165,59	8101,03	7443,83	6889,09	10513,73	8386,25	8420,83	1811,41

Ієрархічні рівні алгоритмів при реалізації операцій над точками еліптичних кривих

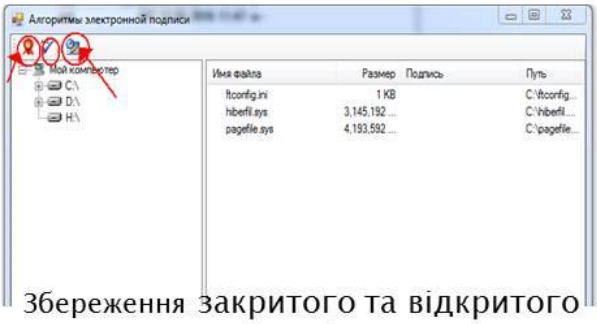


Ієрархія класів, що моделюють скінченні поля

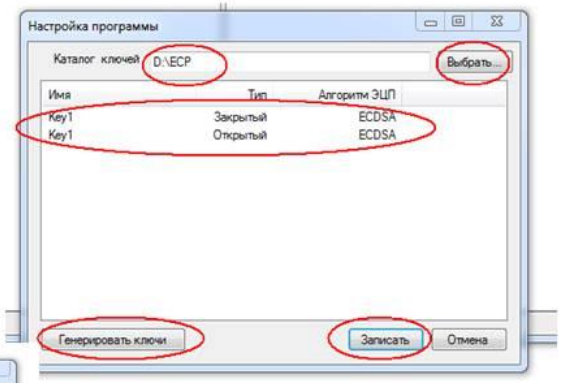


Програмний комплекс

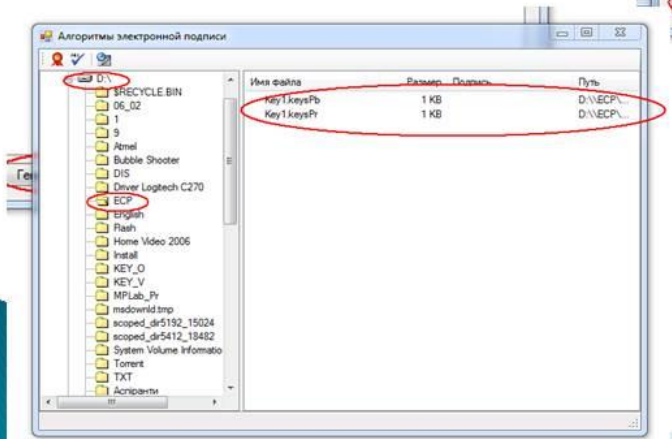
Загальне вікно програми



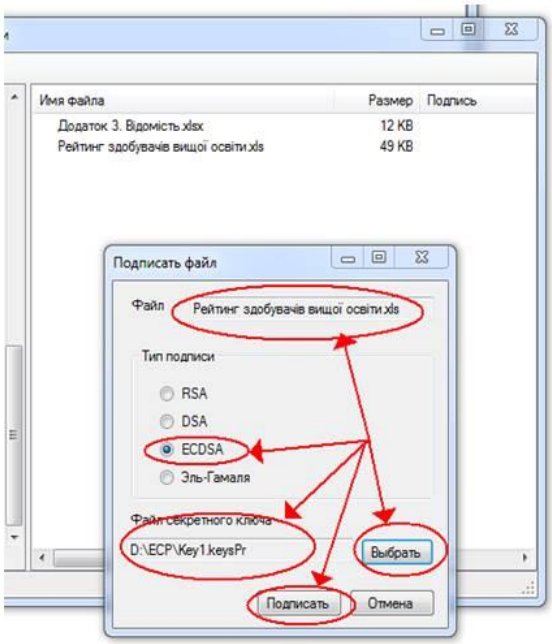
Генерації відкритого та закритого ключів



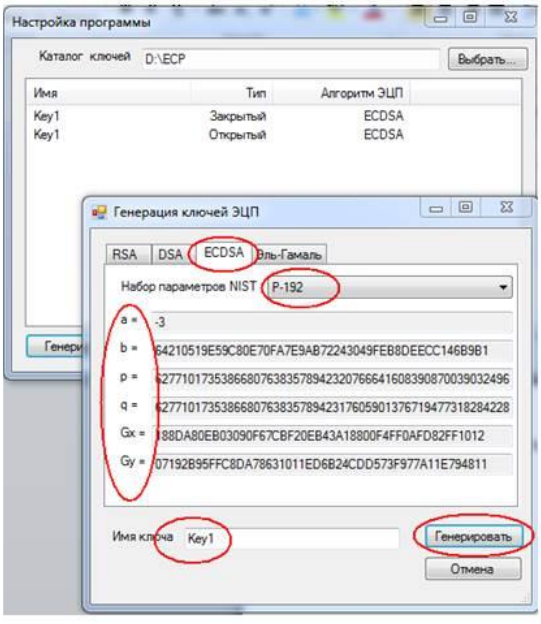
Збереження закритого та відкритого ключів

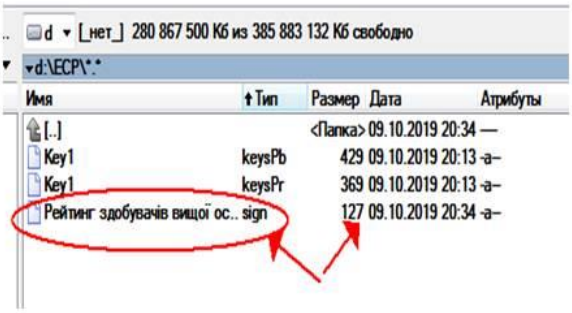


Вибір алгоритму ECDSA та файлу для підпису

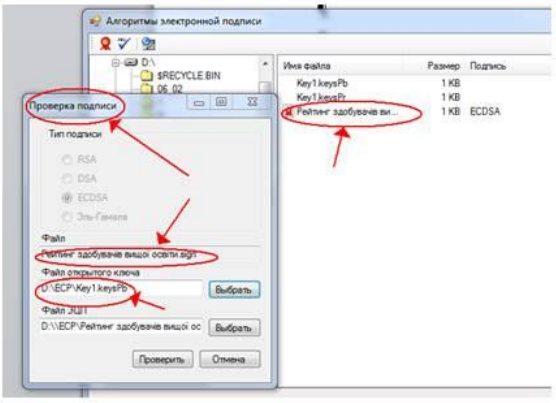


Вибір параметрів алгоритму



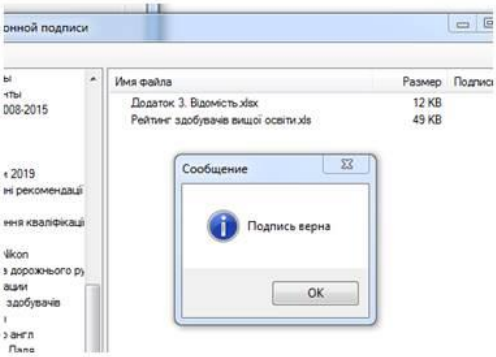


Запис файлу з ЕЦП



Вибір параметрів для перевірки ЕЦП

Завершення перевірки ЕЦП



Наукова новизна та практичне використання

Запропоновано модифікацію методу багатократного скалярного множення точки еліптичної кривої на число, яка полягає у виконанні спеціального передобчислення, і на відміну від існуючих методів дозволяє звести виконання операції множення точки еліптичної кривої на число до додавання, що забезпечує вищу швидкодію (до 25%) порівняно з існуючими для еліптичних кривих з параметрами над GF(p) та $GF(2^m)$.

Розроблено архітектуру програмного забезпечення, особливістю якої є інкапсуляція алгоритму багатократного скалярного множення точки еліптичної кривої на число, що дозволяє на основі шаблону замінювати конкретну реалізацію в екземплярі класу електронно-цифрового підпису без застосування наслідування.

Практичне використання результатів полягає у наступному: **запропоновані** алгоритми дозволяють швидше виконувати операцію багатократного скалярного множення точок еліптичної кривої на число, що дозволяє скоротити час виконання алгоритму електронно-цифрового підпису.

Розроблене математичне та програмне забезпечення для виконання операцій з точками еліптичної кривої дозволяє спростити подальші дослідження у галузі еліптичної криптографії.

Розроблений програмний комплекс може бути використаний в навчальному процесі кафедри комп'ютерних наук та інженерії при проведені практичних занять з дисципліни «Захист інформації в комп'ютерних системах».

Висновки

В даній роботі виконано дослідження підходів та методів шифрування/дешифрування з використанням еліптичних кривих, зокрема алгоритми електронно-цифрового підпису. Розглянуті еліптичні криві, параметри яких визначені над скінченними полями з простою характеристикою $GF(p)$ та бінарним розширенням $GF(2^m)$.

Запропоновано новий метод багатократного скалярного множення точок еліптичної кривої на число, що показує кращу швидкодію в порівнянні з існуючими методами для еліптичних кривих з параметрами, визначеними над полем $GF(2^m)$.

Розроблений програмний комплекс електронно-цифрового підпису з відкритим вихідним кодом з розширюваною програмною архітектурою, що дозволяє динамічно додавати та використовувати різні програмні реалізації методу багатократного скалярного множення точок еліптичної кривої на число, виконувати ЕЦП довільної довжини. Даний комплекс задовольняє основним законам об'єктно-орієнтованого проектування SOLID.