УДК 004.021

Дичка І. А.[1], Онай М. В.[2], Дрозда Т. П.[3]

[1]Професор, доктор технічних наук, декан факультету прикладної математики НТУУ «КПІ», Київ, Україна
[2]Старший викладач кафедри програмного забезпечення комп'ютерних систем факультету прикладної математики НТУУ «КПІ», Київ, Україна
[3]Магістрант кафедри програмного забезпечення комп'ютерних систем факультету прикладної математики НТУУ «КПІ», Київ, Україна

МОДИФІКОВАНИЙ ВІКОННИЙ МЕТОД ОДНОКРАТНОГО МНОЖЕННЯ ТОЧКИ ЕЛІПТИЧНОЇ КРИВОЇ НА СКАЛЯР У ПОЛІ *GF(P)*

При реалізації багатьох криптографічних додатків виникає потреба у швидких алгоритмах множення точки еліптичної кривої на число. У даній статті запропоновано модифікований віконний метод однократного множення точки еліптичної кривої на скаляр у полі GF(p). Об'єктом дослідження є процеси виконання операцій у еліптичних криптосистемах. Предметом дослідження є методи та алгоритми виконання операцій однократного множення точки еліптичної кривої на число у полі GF(p). Метою даного дослідження є розроблення та оптимізація методів і алгоритмів виконання операції множення точки еліптичної кривої на скаляр у полі GF(p) для поліпшення часових характеристик. Існуючі та запропоновані алгоритми реалізовані на мові програмування C# у середовищі розробки Visual Studio 2013. У даній статті проведено дослідження існуючих алгоритмів скалярного множення точки еліптичної кривої та розроблено три модифікації LR-алгоритму віконного методу і узагальнену модифікацію. Експериментальні дослідження реалізованих алгоритмів проводились згідно запропонованої нами методики, яка дозволяє нівелювати вплив на результати дослідження множника та точки еліптичної кривої. Проведене експериментальне дослідження віконних методів та їх модифікацій показало збільшення швидкодії роботи модифікованих алгоритмів у порівнянні з існуючими в середньому на 13%.

Ключові слова: ЕОМ, еліптична криптографія, скалярне множення, таблиця передобчислень, еліптична крива, скінченне поле.

НОМЕНКЛАТУРА

ЕОМ – електронно-обчислювальна машина;

ЕК – еліптична крива;

GF(*p*) – Galois field, поле Галуа або скінченне поле, де p – просте число, що є кількістю елементів поля;

LR – left-to-right;

RL – right-to-left;

NAF – a non-adjacent form;

wNAF – a window non-adjacent form;

p – модуль;

a, b – параметри еліптичної кривої;

x, y – змінні еліптичної кривої;

k – множник, скаляр;

P, Q – точка еліптичної кривої;

w – довжина вікна;

n – бітова довжина скаляра k;

Table – таблиця передобчислень;

for – початок циклу з лічильником;

end for – кінець циклу з лічильником;

while – початок циклу з передуомовою;

end while – кінець циклу з передуомовою;

do – виконати;

to – до;

downto – вниз до;

if – умовний оператор;

then – гілка умовного оператору за якою виконується перехід, якщо умова справедлива;

else – гілка умовного оператору за якою відбувається перехід, якщо умова не виконується;

return – видати результат на вихід алгоритму;

$(k_{l-1},...,k_1,k_0)_r$ – l розрядів подання числа k у системі числення з основою r.

ВСТУП

Задача забезпечення конфіденційності інформації та захист її від зловмисників стає дедалі складнішою і в той же час найбільш актуальною. Більшість систем захисту інформації будується на основі асиметричних криптосистем. Одним з розділів криптографії, який вивчає асиметричні криптосистеми, що засновані на еліптичних кривих над скінченними полями, є еліптична криптографія. Вона бере свій початок ще з 80-х років XX ст. (запропонована Віктором Міллером і Нілом Кобліцем) [1, 2]. Перевага використання еліптичних кривих в криптографічних цілях базується на складності розв'язання задач дискретного логарифмування у групі точок еліптичною кривою. Еліптична криптографія забезпечує набагато вищий рівень криптостійкості використовуючи ключі, що мають меншу довжину у порівнянні з іншими популярними криптосистемами, що засновані на факторизації цілих чисел та проблемі дискретного логарифмування у мультиплікативній групі кільця лишків за певним модулем. У даному дослідженні буде розглянута еліптична крива над полем *GF*(*p*). Елементами цього поля є цілі додатні числа від 0 до $p-1$, де p – модуль. У випадках коли p не дорівнює 2 або 3 буде-яке рівняння еліптичної кривої можна звести до форми Веєрштраса [3]. Тому дана робота ґрунтуватиметься на несингулярній еліптичній кривій у формі Веєрштраса: $y^2 = x^3 + ax + b$, де $p \neq 2,3$.

Найбільш обчислювально витратною операцією в еліптичних криптосистемах є операція множення точки еліптичної кривої на скаляр. У зв'язку з цим актуальною є задача прискорення роботи існуючих алгоритмів скалярного множення точки еліптичної кривої.

1 ПОСТАНОВКА ЗАДАЧІ

Прискорення операції скалярного множення на еліптичній кривій викликає інтерес багатьох дослідників у галузі криптографії [3–9]. З цією метою була запропонована низка методів виконання даної операції [4–9], що в основному полягають в представленні скаляра у деякій з відповідних форм та виконанні різних операцій на еліп-

DOI 10.15588/1607-3274-2016-2-12

тичній кривій (додавання, подвоєння, зменшення у два або три рази).

Множення цілого числа k на точку P еліптичної кривої можна представити наступним чином: $[k]P = \underbrace{P+P+...+P}_{k}$.

Зазвичай число k представляють у двійковому вигляді та за допомогою методів додавання і подвоєння знаходять $[k]P$. Для чисел великих порядків здійснення множення таким чином буде виконуватися занадто довго, тому актуальним є дослідження та модифікація віконних методів, які аналізують одночасно кілька розрядів подання множника у певній системі числення.

Таким чином, метою даного дослідження є модифікація віконного алгоритму виконання операції множення точки еліптичної кривої на скаляр у полі $GF(p)$ для поліпшення часових характеристик алгоритму.

2 ОГЛЯД ЛІТЕРАТУРИ

Вперше ідея подання скаляра у певній системі числення була використана Дональдом Кнутом [3], для мультиплікативної групи, тобто для піднесення до степеня відносно операції множення. Пізніше Хенкерсон та Менезес [4] використали цю ідею для побудови алгоритмів піднесення до степеня в адитивній групі. Скалярне множення точки еліптичної кривої є піднесенням до степеня в адитивній групі точок еліптичної кривої.

Загалом всі алгоритми множення точки еліптичної кривої на число ґрунтуються на поданні множника у певній системі числення та розгляді розрядів цього подання зліва направо, тобто від старшого до молодшого – *LR* або справа наліво, від молодшого до старшого – *RL*.

В літературі [4–8] розглядаються різні модифікації віконних алгоритмів скалярного множення, які завдяки побудові таблиць передобчислень (precomputation table) на початкових стадіях алгоритмів дають хороші показники швидкодії.

Віконні методи отримали таку назву через те, що в них розглядається не по одному двійковому розряду, а по w розрядів, де w – довжина вікна. В роботі [6] Метью Рівайн розглядає два види віконних алгоритмів: *LR*- та *RL*-алгоритми.

Нехай k – деякий скаляр на який виконується множення точки еліптичної кривої, тоді його подання у системі числення за основою 2^w матиме вигляд

$$k = \sum_{i=0}^{l-1} k_i 2^{iw},$$

де $k_i \in \{0,1,...2^w - 1\}$, $k_{l-1} \neq 0$, $l = \left\lfloor \frac{n}{w} \right\rfloor$ та n – бітова довжина скаляра k.

Віконний *LR*-алгоритм полягає у виконанні обчислень за наступними формулами:

$$T_{l-1} = [k_{l-1}]P$$

$$T_i = \left[2^w\right] T_{i+1} + [k_i]P, i = l-2..0.$$

Після проведення обчислень за цими формулами значення $[k]P$ буде знаходитися у T_0.

Віконний *RL*-алгоритм передбачає виконання обчислень за такою формулою:

$$[k]P = \sum_{i=0}^{l-1} \left[k_i \cdot 2^{iw}\right] P.$$

Для обох алгоритмів на початковій стадії будується таблиця передобчислень, яка складається з $2^w - 1$ елементів.

Алгоритм 1 – Бінарний віконний *LR*-алгоритм

Вхід: $P \in E(GF(p)), k = (k_{l-1},...,k_1,k_0)_{2^w} \in \mathbb{N}$

Вихід: $Q = [k]P$

1. *for* $i = 1$ *to* $2^w - 1$ *do*
1.1. $Table[i] \leftarrow i \cdot P$
2. *end for*
3. $Q \leftarrow 0$
4. *for* $i = l-1$ *downto* 0 *do*
4.1. $Q \leftarrow 2^w \cdot Q$
4.2. *if* $k_i > 0$ *then* $Q \leftarrow Q + Table[k_i]$
5. *end for*
6. *return* Q

Алгоритм 2 – Бінарний віконний *RL*-алгоритм

Вхід: $P \in E(GF(p)), k = (k_{l-1},...,k_1,k_0)_{2^w} \in \mathbb{N}$

Вихід: $Q = [k]P$

1. *for* $i = 1$ *to* $2^w - 1$ *do*
1.1. $Table[i] \leftarrow i \cdot P$
2. *end for*
3. $Q \leftarrow 0$
4. *for* $i = 0$ *to* $l-1$ *do*
4.1. *if* $k_i > 0$ *then* $Q \leftarrow Q + Table[k_i]$
4.2. $Table \leftarrow 2^w \cdot Table$
5. *end for*
6. *return* Q

Наприклад, таблиця передобчислень для вікна довжиною 3, тобто $w = 3$ матиме вигляд (P – точка еліптичної кривої):

№ елементу	Значення точки
1	$001 \cdot P$
2	$010 \cdot P$
3	$011 \cdot P$
4	$100 \cdot P$
5	$101 \cdot P$
6	$110 \cdot P$
7	$111 \cdot P$

При аналітичному аналізі алгоритмів 1 та 2 стає зрозумілим, що *RL*-алгоритм буде повільнішим за *LR*-алгоритм, оскільки у *RL*-алгоритмі на кожній ітерації циклу виконується переобчислення значень, що записані у таблицю, тому далі ми будемо розглядати лише *LR*-алгоритми реалізації віконних методів.

Наступним методом скалярного множення точки ЕК, що розглядається в роботі [8] є метод з пересувним вікном. Даний метод дістав таку назву через те, що згідно цього методу необхідно виділяти вікно довжиною w біт тільки якщо старший біт дорівнює 1, в іншому випадку виконують дії передбачені відповідним бінарним алгоритмом.

При розробці віконного методу з пересувним вікном ставилось за мету досягти компромісу між кількістю додавань і подвоєнь точки. На початковій стадії цього методу будується таблиця передобчислень, що містить елементи $[t]P$ для $t = \{2^{w-1}, 2^{w-1}+1, ..., 2^w - 1\}$. *LR*-представлення для скаляра k буде мати наступний вигляд: $k = k_0 + 2k_1 + 2^2 k_2 + ... + 2^m k_m$.

Алгоритм 3 – Бінарний *LR*-алгоритм з пересувним вікном

Вхід: $P \in E(GF(p)), k \in \mathbb{N}$

Вихід: $Q = [k]P$

1. $Q \leftarrow 0, i \leftarrow \log_2 k$
2. *while* $i \geq 0$ *do*
2.1. *if* $(k_i = 0)$ *then* $Q = 2 \cdot Q$
2.2. *else*
2.2.1. *if* $i \geq w - 1$ *then*
a) $t \leftarrow (k_i, ..., k_{i-w+1})$
b) $Q \leftarrow 2^w \cdot Q$
c) $Q \leftarrow Q + t \cdot P$
2.2.2. *else*
a) Викликати бінарний *LR*-алгоритм
2.2.3 $i \leftarrow i - w$
3. *end while*
4. *return* Q

Одним з напрямків прискорення методів скалярного множення точки ЕК є переведення множника у *NAF* представлення [4]. Подання множника k у формі *NAF* виражається формулою $k = \sum_{i=0}^{l-1} k_i 2^i$, де $k_i \in \{0; \pm 1\}$ та $k_{l-1} \neq 0$.

Метод з поданням множника у вигляді *NAF*, є ефективнішим (при наявності *NAF*-розкладення множника) ніж звичайні віконні методи через те, що два сусідні розряди не можуть бути одночасно не нульовим, а це скорочує кількість операцій додавання і віднімання точки. Віконна реалізація даного методу дістала назву метод з поданням множника у вигляді *wNAF*. Подання множника у *wNAF* формі виражається формулою $k = \sum_{i=0}^{n-1} t_i 2^i$, де кожне ненульове t_i є непарним та таким, що $|t_i| < 2^{w-1}$, $t_{n-1} \neq 0$ і хоча б один з w послідовних біт є не нульовим.

На початковій стадії реалізації *LR*-алгоритму цього методу множення потрібно крім побудови таблиці передобчислень перевести скаляр у *wNAF* представлення використовуючи алгоритм 4.

Алгоритм 4 – Переведення додатного цілого числа k у *wNAF* подання

Вхід: позитивне ціле k, ширина вікна w

Вихід: *wNAF(k)*

1. $i \leftarrow 0$
2. *while* $k \geq 1$ *do*
2.1. *if* k є непарним *then*
2.1.1. $t_i \leftarrow k \bmod 2^w$
2.1.2. $k \leftarrow k - t_i$
2.2. *else* $t_i \leftarrow 0$
2.3. $k \leftarrow \frac{k}{2}, i \leftarrow i + 1$
3. *end while*
4. *return* $\{t_{n-1}, t_{n-2}, ..., t_1, t_0\}$

Алгоритм 5 – *LR*-алгоритм з поданням множника у вигляді *wNAF*

Вхід: позитивне ціле k, $P \in E(GF(p))$, ширина вікна w

Вихід: kP

1. Використати алгоритм 4 для обчислення $wNAF(k) = \sum_{i=0}^{n-1} t_i 2^i$
2. *for* $i = 1$ *to* $2^{w-1} - 1$ з кроком 2 *do*
2.1. $Table[i] = i \cdot P$
3. *end for*
4. $Q \leftarrow 0$
5. *for* $i = n - 1$ *downto* 0 *do*
5.1. $Q \leftarrow 2 \cdot Q$
5.2. *if* $t_i \neq 0$ *then*
5.2.1. *if* $t_i > 0$ *then* $Q \leftarrow Q + Table[t_i]$
5.2.2. *else* $Q \leftarrow Q - Table[t_i]$
6. *end for*
7. *return* Q

Таким чином метод з поданням множника у вигляді *wNAF* є аналогом віконного методу, але оперуючи знаком *NAF* подання дозволяє скоротити об'єм необхідної для зберігання таблиці передобчислень пам'яті вдвічі порівняно з класичним віконним методом. Враховуючи особливості *wNAF* подання цілих додатних чисел таблиця передобчислень буде складатися з елементів $[i]P$, де $i = 1, 3, ..., 2^{w-1} - 1$, тобто порівняно з бінарним віконним методом отримуємо скорочення об'єму таблиці передобчислень у чотири рази.

Логічним вдосконаленням методу з поданням множника у вигляді *wNAF* є метод з пересувним вікном та поданням множника у вигляді *NAF*. На відміну від попереднього алгоритму з пересувним вікном, де ми переводимо число k у *wNAF* представлення, у даному алгоритмі потрібно перевести його у *NAF* форму, використовуючи алгоритм 6, а потім виконувати скалярне множення точки еліптичної кривої за алгоритмом 7.

Алгоритм 6 – Переведення додатного цілого числа k у *NAF* подання

Вхід: позитивне ціле k

Вихід: $NAF(k)$

1. $i \leftarrow 0$
2. $w$hile $k \geq 1$ do
2.1. if k is odd then
2.1.2. $k_i \leftarrow 2-(k \bmod 4)$
2.1.3. $k \leftarrow k-k_i$
2.2. $e$lse $k_i \leftarrow 0$
2.3. $k \leftarrow \frac{k}{2}$
2.4. $i \leftarrow i+1$
3. *end while*
4. *return* $\{k_{l-1}, k_{l-2}, ..., k_1, k_0\}$

Алгоритм 7 – *LR*-алгоритм з пересувним вікном та поданням множника у вигляді *NAF*

Вхід: $P \in E\left(GF(p)\right), k \in \mathbb{N}$

Вихід: $[k]P$

1. Використати алгоритм 6 для обчислення $NAF(k) = \sum_{i=0}^{l-1} k_i 2^i$
2. *for* $i = 1$ *to* $(2^w - (-1)^w)/3 - 1$ з кроком 2 *do*
2.1. $Table[i] = i \cdot P$
3. *end for*
4. $Q \leftarrow 0, i \leftarrow l-1$
5. *while* $i \geq 0$ *do*
5.1. *if* $k_i = 0$ *then* $t \leftarrow 1, u \leftarrow 0$
5.1.1. *else* знайти $\max(t) \leq w$ таке що $u \leftarrow (k_i, ..., k_{i-t+1})$ є непарним
5.2. $Q \leftarrow 2^t \cdot Q$
5.3. *if* $u > 0$ *then* $Q \leftarrow Q + Table[u]$
5.4. *else if* $u < 0$ *then* $Q \leftarrow Q - Table[u]$
5.5. $i \leftarrow i - t$
6. *end while*
7. *return* Q

Таблиця передобчислень для цього методу буде містити такі елементи $[i]P$, де $i = 1, 3, ..., \frac{2(2^w - (-1)^w)}{3} - 1$.

Як зазначалося вище, час роботи віконних алгоритмів прискорюється завдяки побудові на першій стадії їх роботи таблиць передобчислень. У таблиці 1 наведено перелік точок, які необхідно обчислити на початковій стадії кожного з розглянутих алгоритмів.

Таблиця 1 – Наперед обчисленні точки для методів скалярного множення

Методи	Значення множника	Кількість наперед обчислених точок
Бінарний віконний метод	$\{1, 2, ..., 2^w - 1\}$	$2^w - 1$
Бінарний метод з пересувним вікном	$\{2^{w-1}, 2^{w-1}+1, ..., 2^w - 1\}$	$2^w - 2^{w-1}$
Метод з поданням множника у вигляді *wNAF*	$\{1, 3, 5, ..., 2^{w-1} - 1\}$	2^{w-2}
Метод з пересувним вікном та поданням множника у вигляді *NAF*	$\{1, 3, ..., \frac{2(2^w - (-1)^w)}{3} - 1\}$	$\frac{1}{3}(2^w - (-1)^w)$

3 МАТЕРІАЛИ ТА МЕТОДИ

Статистично показано, що двійкові подання чисел довжиною понад 100 біт містять довгі послідовності нулів, тому нами запропонована модифікація віконного методу згідно якої буде виділятися вікно зі старшим одинарним бітом та іншими нульовими. Дану модифікацію назвемо модифікація №1. При такому підході таблиця передобчислень буде складатися з таких елементів $\left[2^i\right]P$, де $i = 0, 1, .., w-1$.

Алгоритм 8 – Модифікований віконний *LR*-алгоритм № 1

Вхід: $P \in E\left(GF(p)\right), k \in \mathbb{N}$

Вихід: $[k]P$

1. $Table[0] \leftarrow P$
2. *for* $i = 1$ *to* $w-1$
2.1. $Table[i] \leftarrow 2 \cdot Table[i-1]$
3. *end for*
4. $Q \leftarrow 0; i \leftarrow \log_2 k$
5. *while* $i \geq 0$ *do*
5.1. $Q = 2 \cdot Q$
5.2. *if* $k_i = 1$ *then*
5.2.1. *if* $k_{i-1} = 0$ *then*
a. знайти $\max(t) \leq w$ таке що $k_{i-1} = k_{i-2} = ... = k_{i-t+1} = 0$
b. $Q \leftarrow 2^{t-1} \cdot Q$
c. $Q \leftarrow Q + Table[t-1]$
d. $i \leftarrow i - t$
5.2.2. *else*
a. $Q \leftarrow Q + P$
b. $i \leftarrow i - 1$
6. *end while*
7. *return* Q

Наприклад, таблиця передобчислень для $w = 5$ матиме вигляд (P – точка еліптичної кривої):

№ елементу	Множник	Значення точки
0	2^0	$1 \cdot P$
1	2^1	$10 \cdot P$
2	2^2	$100 \cdot P$
3	2^3	$1000 \cdot P$
4	2^4	$10000 \cdot P$

У випадку якщо множник k складатиметься переважно з нулів, такий спосіб побудови таблиці передобчислень суттєво збільшить швидкодію віконного методу. Але в протилежному випадку, коли біти скаляра будуть тільки одиничні такий метод не дасть хороших результатів. Тому актуальним є побудова нової таблиці множники якої будуть складатись тільки з одиничних біт. Такий метод назвемо модифікація №2. Множник k таблиці передобчислень в даному методі буде рівним $2^i - 1$, де $i = 1, 2, .., w$.

Алгоритм 9 – Модифікований віконний *LR*-алгоритм № 2

Вхід: $P \in E\left(GF(p)\right), k \in \mathbb{N}$

Вихід: $[k]P$

1. *for* $i = 1$ *to* w
1.1. $Table[i] \leftarrow (2^i - 1) \cdot P$
2. *end for*
3. $Q \leftarrow 0; i \leftarrow \log_2 k$
4. *while* $i \geq 0$ *do*
4.1. $Q = 2 \cdot Q$
4.2. *if* $k_i = 1$ *then*
4.2.1. знайти $\max(t) \leq w$ таке що $k_{i-1} = k_{i-2} = ... = k_{i-t+1} = 1$
4.2.2. $Q \leftarrow 2^{t-1} \cdot Q$
4.2.3. $Q \leftarrow Q + Table[t-1]$
4.2.4. $i \leftarrow i - t + 1$
4.3. $i \leftarrow i - 1$
5. *end while*
6. *return Q*

Для вікна $w = 5$ таблиця пердобчислень матиме вигляд:

8	Множник	Значення точки
1	$2^1 - 1$	$1 \cdot P$
2	$2^2 - 1$	$11 \cdot P$
3	$2^3 - 1$	$111 \cdot P$
4	$2^4 - 1$	$1111 \cdot P$
5	$2^5 - 1$	$11111 \cdot P$

Також можна сподіватися на приріст швидкодії, коли з розрядів числа k буде виділятися частина біт, що починається і закінчується одиницею, а між ними міститиметься певна кількість нулів (модифікація №3). Множник k у таблиці передобчислень в модифікації №3 буде рівним $2^{i-1} + 1$, де $i = 2, 3, .., w$.

Алгоритм 10 – Модифікований віконний *LR*-алгоритм № 3

Вхід: $P \in E\left(GF(p)\right), k \in \mathbb{N}$

Вихід: $[k]P$

1. $Table[1] \leftarrow P$
2. *for* $i = 2$ *to* w
2.1. $Table[i] \leftarrow (2^{i-1} + 1) \cdot P$
3. *end for*
4. $Q \leftarrow 0; i \leftarrow \log_2 k$
5. *while* $i \geq 0$ *do*
5.1. $Q = 2 \cdot Q$
5.2. *if* $k_i = 1$ *then*
5.2.1. *if* $k_{i-1} = 0$ *then*
a. знайти $\max(t) \leq w$ таке що $k_i = k_{i-t+1} = 1$ та $k_{i-1} = k_{i-2} = ... = k_{i-t+2} = 0$
b. $Q \leftarrow 2^{t-1} \cdot Q$
c. $Q \leftarrow Q + Table[t]$
d. $i \leftarrow i - t + 1$
5.2.2. *else*
a. $Q \leftarrow 2 \cdot Q$
b. $Q \leftarrow Q + Table[2]$
c. $i \leftarrow i - 1$
5.2.3. $i \leftarrow i - 1$
6. *end while*
7. *return Q*

Оскільки за допомогою виразу $2^{i-1} + 1$ неможливо отримати 1, то до початку циклу побудови таблиці передобчислень у елемент таблиці з індексом 1, необхідно записати значення точки P. Приклад такої таблиці пердобчислень для вікна $w = 5$:

№ елементу	Множник	Значення точки
1	1	$1 \cdot P$
2	$2^1 + 1$	$11 \cdot P$
3	$2^2 + 1$	$101 \cdot P$
4	$2^3 + 1$	$1001 \cdot P$
5	$2^4 + 1$	$10001 \cdot P$

У таблиці 2 наведено значення множників для таблиць передобчислень запропонованих модифікацій. Зрозуміло, що кожна із запропонованих модифікацій в певному частковому випадку буде збільшувати швидкодію віконного алгоритму скалярного множення.

Модифікації № 1–3 дають приріст швидкодії тільки в окремих випадках, тому актуальним є побудувати узагальнений модифікований віконний метод скалярного множення точки ЕК та об'єднати переваги кожної із запропонованих модифікацій таким чином, щоб прискорити віконний алгоритм для будь-якого вигляду двійкового представлення множника k. Доцільним буде об'єднувати таблиці передобчислень з модифікації №2 та №3, оскільки модифікація №1 покривається модифікацією №3. Особливістю побудови таблиць передобчислень для другої та третьої модифікації є те, що номер елемента у таблиці відповідає бітовій довжині множника для якого обчислене відповідне значення.

Алгоритм 11 – Узагальнений модифікований віконний *LR*-алгоритм скалярного множення

Вхід: $P \in E\left(GF(p)\right), k \in \mathbb{N}$

Вихід: $[k]P$

1. $Table1[1] \leftarrow P$
2. $Table2[1] \leftarrow P$
3. *for* $i = 2$ *to* w *do*
3.1. $Table1[i] \leftarrow (2^i - 1) \cdot P$
3.2. $Table2[i] \leftarrow (2^{i-1} + 1) \cdot P$
4. *end for*
5. $Q \leftarrow 0, i \leftarrow \log_2 k$

Таблиця 2 – Наперед обчислені точки для модифікованих віконних методів

Метод	Значення множника	Кількість наперед обчислених точок
Модифікація №1	2^i, де $i \in [0; w-1]$	w
Модифікація №2	$2^i - 1$, де $i \in [1; w]$	w
Модифікація №3	1 та $2^{i-1} + 1$, де $i \in [2; w]$	w

6. *while* $i \geq 0$ *do*
6.1. *if* $k_i = 1$ *then*
6.1.1. *if* $k_{i-1} = 0$ *then*
a. знайти $\max(t) \leq w$ такий, що $k_i = k_{i-t+1} = 1$ *and* $k_{i-1} = k_{i-2} = ... = k_{i-t+2} = 0$
b. $Q \leftarrow 2^t \cdot Q$
c. $Q \leftarrow Q + Table1[t]$
6.1.2. *else*
a. знайти $\max(t) \leq w$ такий, що $k_{i-1} = k_{i-2} = ... = k_{i-t+1} = 1$
b. $Q \leftarrow 2^t \cdot Q$
c. $Q \leftarrow Q + Table2[t]$
6.1.3. $i \leftarrow i - t$
6.2. *else*
6.2.1. $Q = 2 \cdot Q$
6.2.2. $i = i - 1$
7. *end while*
8. *return Q*

На кроці 11 при реалізації алгоритму 11 потрібно передбачити перевірку чи є праворуч від *i*-го біта одиничні біти. Якщо всі біти молодші *i*-го є нульовими, то необхідно виконати такі дії: $Q = 2 \cdot Q$, $Q = Q + P$, $i = i - 1$ та перейти на наступну ітерацію циклу *while*.

4 ЕКСПЕРИМЕНТИ

З метою проведення експериментальних досліджень було розроблено програмний продукт на мові програмування C# у середовищі розробки «Visual Studio 2013». Експериментальне дослідження проводилося на ЕОМ з операційною системою Windows 8.1, об'ємом оперативної пам'яті 2Gb, процесором Pentium Dual-Core 2,30 Hz.

Даний програмний продукт дозволяє проводити тестування коректності роботи алгоритмів та проводити дослідження розглянутих віконних методів скалярного множення точки ЕК. Завдяки тестуванню віконних алгоритмів для різних значень довжини вікна *w* було отримано оптимальне значення *w*.

У розробленому програмному продукті, окрім відомих алгоритмів реалізовано запропоновані нами модифікації віконних *LR*-алгоритмів, оскільки, як показало дослідження *LR*-алгоритми показують кращі результати ніж *RL*-алгоритми. Для збільшення швидкодії алгоритмів на початковій стадії роботи кожного з них будується таблиця передобчислень, значення таблиць передобчислень наведені у таблиці 1 та таблиці 2. При аналізі літературних джерел [5, 7] було встановлено, що оптимальними параметрами еліптичної кривої, які забезпечують високу крипостійкість є: $a = 79, b = -3$. Тому ці значення параметрів було використано у дослідженні.

Замір швидкодії роботи алгоритмів проводився за наступною методикою:

1) обирається довжина модуля *p* (64, 128, 256 та 512 біт);

2) для обраного модуля *p* формується множина з 25 випадкових точок еліптичної кривої;

3) випадковим чином генерується 25 множників довжиною від 32 до 64 біт;

4) виконується множення кожної точки з п. 2 на кожен множник з п. 3, отримані часові показники усереднюються.

Пошук оптимальних значень довжини вікна (рис. 1) виконувався для модуля *p* що має довжину 128 біт та довжин вікна $w \in [2;9]$, оскільки після збільшення довжини вікна до 10 біт час виконання алгоритмів зростає. Значення множників змінювалось від 10 до 1000 з кроком 100, оскільки для проведення дослідження при більших значеннях множника було не достатньо обчислювальної потужності комп'ютера.

5 РЕЗУЛЬТАТИ

За наведеною методикою для існуючих віконних *LR*- та *RL*-алгоритмів було побудовано таблицю 3 та для запропонованих модифікацій і узагальненого модифікованого віконного методу таблицю 4.

Як можна помітити з таблиці 3, *LR*-алгоритми дають значно кращі результати ніж *RL*- алгоритми, що підтверджує результати проведеного аналітичного дослідження, тому доцільним є пошук оптимального значення довжини вікна *w* для існуючих *LR*-алгоритмів.

На рис. 1 наведено залежність часу роботи *LR*-алгоритмів від довжини вікна, де 1, 3, 5, 7 – номери відповідних алгоритмів у таблиці 3.

Як видно з рисунку 1 найкращі часові характеристики показують алгоритми 1 і 3, а саме бінарний віконний *LR*-алгоритм та бінарний *LR*-алгоритм з пересувним вікном при довжині вікна 9 біт, що підтверджує доцільність їх модифікацій. Швидкодія роботи модифікованих віконних *LR*-алгоритмів та узагальненого алгоритму наведена в таблиці 4.

Таблиця 3 – Часові характеристики методів скалярного множення, мс

№	Методи	Довжина модуля, біт			
		64	128	256	512
1	Бінарний віконний *LR*-алгоритм	5,5	9,0	32,1	84,4
2	Бінарний віконний *RL*-алгоритм	54,7	120,8	403,6	1125,7
3	Бінарний *LR*-алгоритм з пересувним вікном	6,8	10,3	32,5	84,6
4	Бінарний *RL*-алгоритм з пересувним вікном	33,8	73,7	257,0	696,2
5	*LR*-алгоритм з пересувним вікном та поданням множника у вигляді *NAF*	4,3	8,6	30,5	84,2
6	*RL*-алгоритм з пересувним вікном та поданням множника у вигляді *NAF*	19,1	42,1	144,7	390,8
7	*LR*-алгоритм з поданням множника у вигляді *wNAF*	4,1	8,9	32,1	87,0
8	*RL*-алгоритм з поданням множника у вигляді *wNAF*	14,7	33,1	110,8	311,1

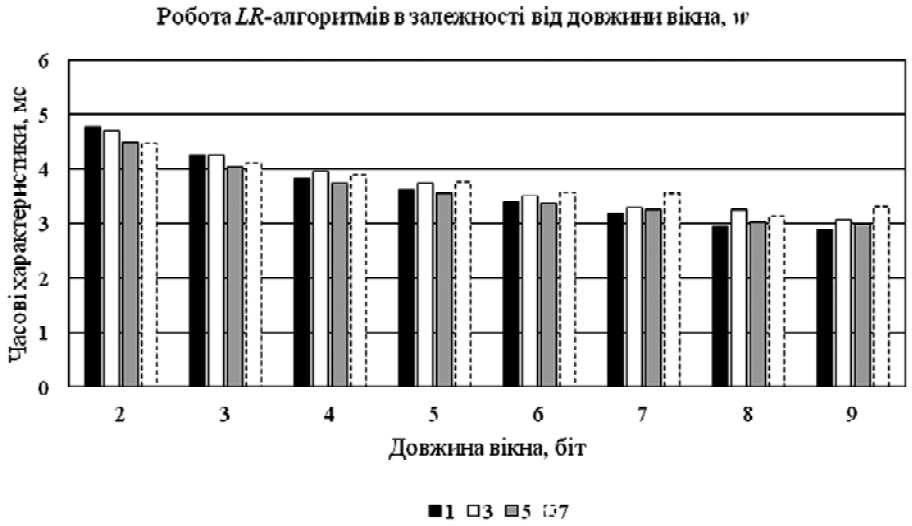

Рисунок 1 – Часові характеристики віконних *LR*-алгоритмів в залежності від довжини вікна

Таблиця 4 – Часові характеристики модифікованих методів в залежності від довжини модуля

№	Метод	Довжина модуля, біт			
		64	128	256	512
1	Модифікація №1	4,7	11,1	39,1	106,6
2	Модифікація №2	4,3	9,2	34,4	91,1
3	Модифікація №3	4,3	9,0	31,2	86,4
4	Узагальнений модифікований віконний метод	4,1	8,2	31,1	82,0

6 ОБГОВОРЕННЯ

За результатами наведеними у таблиці 3, найкращі часові характеристики для модулів будь-якої довжини показують *LR*-алгоритм з поданням множника у вигляді *wNAF* та бінарний *LR*-алгоритм з пересувним вікном і поданням множника у вигляді *NAF*. При правильно підібраній довжині вікна (рис. 1) та довжині модуля 128 біт найкращі часові характеристики показують віконний *LR*-алгоритм та *LR*-алгоритм з пересувним вікном. З таблиці 4 видно, що побудовані модифікації методу з пересувним вікном дають приріст швидкодії для модулів довжиною 64 біт на 20%. Модифікація №3 порівняно з попередніми модифікаціями покращує часові характеристики майже на 3% для модулів довжиною 128 і 256 біт.

Найефективнішою є побудова узагальненої модифікації віконного методу та використання таблиць передобчислень зі значеннями $[t]P$ для $t = 2^i + 1$, де $i = 1..w$ та $t = 2^i - 1$, де $i = 2..w$. Побудований *LR*-алгоритм на її основі дає приріст швидкодії в середньому на 13% для модулів 64, 128, 256 та 512 біт. Запропоновані таблиці передобчислень використовують порівняно з бінарними віконними методами у $\frac{2^w - 1}{w}$ разів менше оперативної пам'яті.

ВИСНОВКИ

У роботі вирішено актуальну задачу вдосконалення існуючих методів скалярного множення точки еліптичної кривої у полі *GF*(*p*).

Наукова новизна роботи полягає у тому, що дістав подальшого розвитку науковий підхід модифікації бінарних методів множення точки еліптичної кривої на число, що заснований на розгляді кількох розрядів двійкового подання множника одночасно на кожній ітерації циклу.

Проведений аналіз віконних методів множення точки еліптичної кривої на скаляр у полі *GF*(*p*) показав, що при правильному виборі довжини вікна та побудові на початковій стадії таблиць передобчислень можна суттєво збільшувати швидкодію алгоритмів. За допомогою розробленого програмного забезпечення для аналізу і тестування методів множення точки ЕК на скаляр, проведено експериментальне дослідження, яке довело практичну доцільність використання запропонованих методів, замість існуючих.

Розроблені три модифікації віконного *LR*-алгоритму з пересувним вікном, які відрізняються від існуючого методу способом побудови таблиць передобчислень, забезпечують приріст швидкодії для множників спеціальної структури. Побудована узагальнена модифікація віконного *LR*-алгоритму з пересувним вікном, що полягає у комбінації, на початковій стадії роботи алгоритму, двох таблиць передобчислень з модифікації №2 та №3, яка забезпечує приріст швидкодії порівняно з існуючими методами в середньому на 13%.

Практична цінність отриманих результатів полягає у тому, що розроблено програмне забезпечення, яке реалізує запропоновані модифікації та існуючі методи множення точки еліптичної кривої на число і дозволяє проводити аналіз алгоритмів, що реалізують зазначені методи. За допомогою даного програмного забезпечення може бути вирішена практична задача вибору найкращого алгоритму скалярного множення точки еліптичної кривої для використання у алгоритмі цифрового підпису на еліптичних кривих, який широко використовується.

Перспективою для подальшого дослідження є побудова модифікованого методу з пересувним вікном та поданням множника у вигляді *NAF* та модифікованого методу з поданням множника у вигляді *wNAF*.

ПОДЯКИ

Дослідження виконано у межах держбюджетної науково-дослідної теми «Розроблення та дослідження високоефективних архітектур спеціалізованих комп'ютерних систем для реалізації обчислень у скінченних полях» Національного технічного університету України «Київський політехнічний інститут» (номер державної реєстрації 0115U000319).

СПИСОК ЛІТЕРАТУРИ

1. Miller V. Use of elliptic curves in cryptography / V. Miller // Lecture Notes in Computer Science. Advances in cryptology – CRYPTO 85. – Springer, 1986. – P. 417–426. 10.1007/3-540-39799-X_31
2. Koblitz N. Introduction to Elliptic Curves and Modular Forms / Neal Koblitz. – New York : Springer, 1984. – 248 p. 10.1007/978-1-4684-0255-1
3. Knuth, D. The Art of Computer Programming. Volume 2 Seminumerical Algorithms, Third Edition / D. E. Knuth. – Massachusetts: Addison-Wesley, 1997. – 762 p.
4. Hankerson D. Guide to Elliptic Curve Cryptography / D. Hankerson, A. Menezes, S. Vanstone. – New York : Springer, 2004. – 311 p.
5. Crandall R. Prime Numbers. A Computational Perspective. Second Edition / Richard Crandall, Carl Pomerance. – New York : Springer, 2005. – 604 p. 10.1007/978-1-4684-9316-0
6. Rivain M. Fast and Regular Algorithms for Scalar Multiplication over Elliptic Curves / Matthieu Rivain. – IACR Cryptology ePrint Archive, 2011. – 338 p.
7. Болотов, А. А. Алгоритмические основы эллиптической криптографии / А. А. Болотов. – М. : Изд-во, 2004. – 499 с.
8. Elliptic Curve Point Multiplication [Electronic resource] // December 31, 2015: Proceedings. – Mode of access: https://en.wikipedia.org/wiki/Elliptic_curve_point_multiplication. – Last access: 2016. – Title from the screen.
9. Pathak H. Speeding Up Computation of Scalar Multiplication in Elliptic Curve Cryptosystem / H. Pathak, M. Sanghi // International Journal on Computer Science and Engineering. – 2010. – No. 04. – P. 1024–1028.

Стаття надійшла до редакції 15.02.2016.
Після доробки 01.03.2016.

Дичка И. А.[1], Онай Н. В.[2], Дрозда Т. П.[3]

[1]Профессор, доктор технических наук, декан факультета прикладной математики НТУУ «КПИ», Киев, Украина

[2]Старший преподаватель кафедры программного обеспечения компьютерных систем факультета прикладной математики НТУУ «КПИ», Киев, Украина

[3]Магистрант кафедры программного обеспечения компьютерных систем факультета прикладной математики НТУУ «КПИ», Киев, Украина

МОДИФИЦИРОВАННЫЙ ОКОННЫЙ МЕТОД ОДНОКРАТНОГО УМНОЖЕНИЯ ТОЧКИ ЭЛЛИПТИЧЕСКОЙ КРИВОЙ НА СКАЛЯР В ПОЛЕ GF(P)

При реализации многих криптографических приложений возникает потребность в быстрых алгоритмах умножения точки эллиптической кривой на число. В данной статье предложен модифицированный оконный метод однократного умножения точки эллиптической кривой на скаляр над полем GF(p). Объектом исследования являются процессы выполнения операций в эллиптических криптосистемах. Предметом исследования являются методы и алгоритмы выполнения операций однократного умножения точки эллиптической кривой на число над полем GF(p). Целью данного исследования является разработка и оптимизация методов и алгоритмов выполнения операции умножения точки эллиптической кривой на скаляр над полем GF(p) для улучшения временных характеристик. Существующие и предложенные алгоритмы реализованы на языке программирования C# в среде разработки Visual Studio 2013. В данной статье проведено исследование существующих алгоритмов скалярного умножения точки эллиптической кривой и разработаны три модификации LR-алгоритма оконного метода и обобщенная модификация. Экспериментальные исследования реализованных алгоритмов проводились согласно предложенной нами методики, которая позволяет нивелировать влияние на результаты исследования множителя и точки эллиптической кривой. Проведенное экспериментальное исследование оконных методов и их модификаций показало увеличение быстродействия работы модифицированных алгоритмов по сравнению с существующими в среднем на 13%.

Ключевые слова: ЭВМ, эллиптическая криптография, скалярное умножение, таблица предвычислений, эллиптическая кривая, конечное поле.

Dychka I. A., Onai M. V., Drozda T. P.

[1]Professor, Dc.Sc., Dean of the Faculty of Applied Mathematics NTUU «KPI», Kyiv, Ukraine

[2]Senior Lecturer, Department of Computer Systems Software of Faculty of Applied Mathematics, NTUU «KPI», Kyiv, Ukraine

[3]Master student, Department of Computer Systems Software of Faculty of Applied Mathematics, NTUU «KPI», Kyiv, Ukraine

MODIFIED METHOD FOR ELIPTIC CURVE SCALAR POINT MULTIPLICATION OVER GF(P)

During development of many cryptographic applications, we need to perform fast algorithms of scalar multiplication. In this paper we propose a modified window method of elliptic curve point multiplication over the GF(p). The object of the research are the processes of performing operations in elliptic cryptosystems. The subject of the research are the methods and the algorithms of elliptic curve point multiplication over the GF(p). The goal of the research is to develop and optimize the methods and the algorithms of performing elliptic curve point multiplication operation over the GF(p) for improving the time characteristics. Existing and proposed algorithms were implemented with C# programming language and integrated development environment – Visual Studio 2013. In this article we did an investigation of the existing algorithms of elliptic curve point multiplication and developed three versions of the window method LR-algorithm and generalized modification. Experimental studies of the implemented algorithms were performed according to the proposed methodology, which allows us to explore the impact of the multiplier and elliptical curve point on the results of the research. The experimental research of window methods and their modifications showed an increase speed of the modified algorithms compared to the existing algorithms in average of 13%.

Keywords: computers, elliptic curve cryptography, scalar multiplication, precomputation table, elliptic curve, finite field.

REFERENCES

1. Miller V. Use of elliptic curves in cryptography, *Lecture Notes in Computer Science. Advances in cryptology – CRYPTO 85*. Springer, 1986, pp. 417–426 10.1007/3-540-39799-X_31
2. Koblitz N. Introduction to Elliptic Curves and Modular Forms. New York, Springer, 1984, 248 p. 10.1007/978-1-4684-0255-1
3. Knuth D. The Art of Computer Programming. Volume 2 Seminumerical Algorithms, Third Edition. Massachusetts, Addison-Wesley, 1997, 762 p.
4. Hankerson D., Menezes A., Vanstone S. Guide to Elliptic Curve Cryptography. New York, Springer, 2004, 311 p.
5. Crandall R., Pomerance C. Prime Numbers. A Computational Perspective. Second Edition. New York : Springer, 2005, 604 p. 10.1007/978-1-4684-9316-0
6. Rivain M. Fast and Regular Algorithms for Scalar Multiplication over Elliptic Curves. IACR Cryptology ePrint Archive, 2011, 338 p.
7. Bolotov A. A. Algoritmicheskie osnovy e'llipticheskoj kriptografii. Moscow, Izd-vo, 2004, 499 p
8. Elliptic Curve Point Multiplication [Electronic resource]. December 31, 2015: Proceedings. Mode of access: https://en.wikipedia.org/wiki/Elliptic_curve_point_multiplication, Last access: 2016, Title from the screen.
9. Pathak H., Sanghi M. Speeding Up Computation of Scalar Multiplication in Elliptic Curve Cryptosystem, *International Journal on Computer Science and Engineering*, 2010, No. 04, pp. 1024–1028.