НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

«На правах рукопису»
УДК 004.052.42

До захисту допущено:
в.о. завідувача кафедри
_____ Михайло НОВОТАРСЬКИЙ
“ ” _____ 2025 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою “Комп’ютерні системи та мережі”
спеціальності 123 “Комп’ютерна інженерія”

на тему: *Метод безпечного залучення хмарних систем для прискореної реалізації криптографії з відкритим ключем в Інтернеті Речей*

Виконала:
студентка ІІ курсу, групи ІО-31мн
Гайдукевич Марія Андріївна _____

Керівник:
доц. каф ОТ, к.т.н., доцент
Марковський Олександр Петрович _____

Консультант з нормоконтролю:
проф. каф. ОТ, д.т.н., професор
Кулаков Юрій Олексійович _____

Рецензент:
декан ФПМ, д.т.н., професор
Дичка Іван Андрійович _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цьому звіті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2025

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Рівень вищої освіти – другий (магістерський)

Спеціальність – 123 “Комп’ютерна інженерія”

Освітньо-наукова програма “Комп’ютерні системи та мережі”

ЗАТВЕРДЖУЮ
в.о. завідувача кафедри
_____ Михайло НОВОТАРСЬКИЙ
“ ” _____ 2025 р

ЗАВДАННЯ
на магістерську дисертацію студентки

Гайдукевич Марії Андріївни

1. Тема дисертації *Метод безпечного залучення хмарних систем для прискореної реалізації криптографії з відкритим ключем в Інтернеті Речей*

Науковий керівник дисертації Марковський Олександр Петрович, к.т.н., доцент кафедри ОТ_____
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від _13 березня 2025 р.__ № 1128-с

2. Термін подання студентом дисертації 05 травня 2025 р.

3. Об’єкт дослідження: процеси захищеного розподіленого обчислення модулярної експоненти – базової операції криптографії з відкритим ключем – із залученням хмарних систем.

4. Предмет дослідження методи прискорення реалізації базової операції криптографії з відкритим ключем – модулярного експоненціювання за рахунок залучення хмарних систем.

5. Перелік завдань, які потрібно розробити – аналіз проблеми та існуючих рішень; розробка моделі/методу/алгоритму/програмного забезпечення; дослідження ефективності розробленого методу/алгоритму/програмного

забезпечення.

6. Консультанти розділів дисертації

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Нормоконтроль	Кулаков Ю.М., професор		

7. Дата видачі завдання _13.10.2024__

Календарний план

№ п/п	Найменування етапів дипломного проєкту	Терміни виконання етапів проєкту	Примітки
1.	*Затвердження теми дисертації*	*26.09.2024-16.10.2024*	
2.	*Вивчення та аналіз завдання*	*16.10.2024-16.11.2025*	
4.	*Розробка алгоритму гомоморфного шифрування для захищеного хмарного обчислення модулярної експоненти*	*16.11.2024-25.02.2025*	
5.	*Програмна реалізація системи*	*25.02.2025-20.03.2025*	
6.	*Оформлення пояснювальної записки*	*20.03.2025-21.04.2025*	
7.	*Проходження нормоконтролю*	*21.04.2025-15.05.2025*	
8.	*Захист*	*15.05.2025*	

Студент __Марія ГАЙДУКЕВИЧ__ _____
(підпис)

Науковий керівник дисертації _Олександр МАРКОВСЬКИЙ_ _____
(підпис)

Анотація до магістерської дисертації
студентки групи ІО-31мн,
Гайдукевич Марії Андріївни

РЕФЕРАТ

на магістерську дисертацію

виконану на тему: Метод безпечного залучення хмарних систем для прискореної реалізації криптографії з відкритим ключем в Інтернеті Речей

студенткою: Гайдукевич Марією Андріївною

Робота складається із вступу та 4 розділів. Загальний об’єм роботи: 95 аркушів основного тексту, 6 ілюстрацій, 4 таблиць та 1 додаток. Для виконання магістерської дисертації було використано інформацію з 62 літературних джерел.

Актуальність теми:

Перманетний прогрес інтернет-технологій призвів до появи широкого класу систем, що забезпечують дистанційне керування об’єктами реального світу. Ці рішення, відомі як Інтернет Речей (Internet of Things), активно впроваджуються в різноманітні галузі людської діяльності. Однією з ключових вимог до IoT-систем є забезпечення високого рівня інформаційної безпеки, оскільки передача даних зазвичай здійснюється через потенційно незахищене середовище – відкриту мережу Інтернет. Для досягнення належного рівня захисту необхідне повноцінне використання наявних криптографічних засобів, зокрема протоколів, орієнтованих на забезпечення конфіденційності, цілісності та автентифікації.

Більшість сучасних протоколів інформаційного захисту базуються на використанні криптографії з відкритим ключем. Основною операцією у такій криптографії виступає модулярне експоненціювання, що виконується над числами великої розрядності. На практиці для забезпечення належного рівня

захищеності застосовують числа розрядністю 4096 біт. Обробка чисел такої довжини вимагає значних обчислювальних ресурсів, що є критичним для термінальних мікроконтролерів з обмеженими можливостями, які використовуються в IoT-пристроях.

Останнім часом один із найбільш перспективних підходів до вирішення проблем обмеженості апаратного забезпечення полягає у залученні хмарних обчислювальних систем. Такий підхід дозволяє використовувати ресурси віддалених систем з високою продуктивністю, включаючи криптопроцесори, що спеціалізуються на виконанні операції модулярного експоненціювання над довгими числами. Однак пряме делегування таких обчислень віддаленій системі може становити загрозу безпеці, оскільки операндами модулярного експоненціювання в контексті криптографії з відкритим ключем виступають зокрема секретні компоненти. Це зумовлює необхідність створення таких методів розподіленого модулярного експоненціювання, які унеможливлюють несанкціоноване відновлення секретних компонентів за даними, що передаються у хмарні системи.

Мета і задачі дослідження:

Мета досліджень полягає у підвищенні ефективності обчислення модулярної експоненти з залученням хмарних технологій за рахунок прискорення обробки частини розрядів коду експоненти на термінальних платформах IoT.

Для досягнення поставленої мети вирішуються наступні задачі:

1. Аналіз проблеми реалізації криптографічних механізмів захисту даних на термінальних мікроконтролерах систем віддаленого управління на базі технології IoT. Огляд існуючих підходів до організації безпечного розподіленого обчислення базової операції криптографії з відкритим ключем – модулярного експоненціювання.

2. Теоретичне обґрунтування, розробка та дослідження методу захищеного обчислення модулярної експоненти з залученням хмарних обчислень в системах моніторингу та управління з передачею проміжних даних від віддаленої системи до термінального мікроконтролеру. Теоретичне та експериментальне дослідження ефективності методу.

3. Розробка математичної моделі розподіленого обчислення модулярної експоненти на основі мультиплікативно-адитивних розкладень коду експоненти.

4. Теоретичне обґрунтування, розробка та дослідження методу розподіленого захищеного обчислення модулярної експоненти на основі багаторівневих мультиплікативно-адитивних розкладень коду експоненти. Теоретичне та експериментальне дослідження ефективності методу.

5. Розробка програмних засобів реалізації запропонованих методів для експериментального дослідження їх ефективності.

Об'єкт дослідження: процеси захищеного розподіленого обчислення модулярної експоненти – базової операції криптографії з відкритим ключем – із залученням хмарних систем.

Предмет дослідження: методи прискорення реалізації базової операції криптографії з відкритим ключем – модулярного експоненціювання за рахунок залучення хмарних систем.

Методи дослідження: Для досягнення поставлених в магістерській дисертації задач, використано положення та методи теорії обчислень, теорії прикладної криптографії, теорії чисел, теорії ймовірностей, статистичного аналізу, а також методи імітаційного моделювання.

Наукова новизна одержаних результатів:

1. Запропоновано метод розподіленого модулярного експоненціювання на термінальних мікроконтролерах IoT з захищеним залученням віддалених

комп’ютерних систем, який відрізняється тим, що послідовності операцій модулярного множення, що співвідносяться з розрядами коду експоненти, які оброблюються на термінальному мікроконтролері заміняється вибором із всіх можливих результатів, обчислених на віддаленій системі, за рахунок чого зменшується обчислювальне навантаження на нього і, відповідно, досягається прискорення розподіленого обчислення модулярної експоненти.

2. Теоретично обґрунтовано, розроблено та дослідження методу прискореної реалізації цієї операції на термінальних пристроях IoT з безпечним використанням хмарних обчислень, відмінність якого полягає у організації багатокомпонентного мультиплікативно-адитивного розкладення коду експоненти, що дозволяє збільшити об’єм перебору, який виконується для незаконної реконструкції секретних компонентів модулярного експоненціювання, і тим самим підвищити рівень захищеності віддаленої реалізації цієї операції.

Практичне значення одержаних результатів:

Практичне значення отриманих результатів визначається тим, що їх використання дозволяє на порядки прискорити реалізацію на термінальних мікроконтролерах систем управління на базі IoT базової операції криптографії з відкритим ключем – модулярного експоненціювання. Тим самим значною мірою знімаються часові обмеження на реалізацій функцій захисту від стороннього втручання в процеси управління. Це дозволяє значно розширити сферу застосування технологій IoT для побудови систем віддаленого контролю і управління об’єктами реального світу в режимі реального часу.

Апробація результатів магістерської дисертації:

Основні результати магістерської дисертації доповідались та обговорювались на міжнародних науково-технічних конференціях

1. Intelligence: proceedings of the International Conference ICSFTI2023, Kyiv, Ukraine, June 29-30 2023. м. Київ.

2. The 14th International Conference on Dependable Systems, Services and Technologies (DESSERT'2024). 12-15 October, Greece, Athens. -2024 (https://www.dessert-conf.org/dessert-2024/)

Публікації:

За темою дисертації опубліковано чотири статті, одна з яких реферується Scopus, а дві опубліковані в фахових виданнях України.

1. Haidukevych Mariia. A Secure Cloud Computing Approach for Rapid Implementation of Public Key Cryptography on IoT Terminal Devices/ Markovskyi Oleksandr, Haidukevych Mariia , José Borges, Nazar Serhiichuk// Proceeding of The 14th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2024 11-13 October, 2024, Athens, Greece. – IEEE publication. – 2024. – pp. 51-54. doi: 10.1109/DESSERT61349.2024.10416477.
2. Haidukevych M. Fast secure calculation of the open key cryptography procedures for iot in clouds / A. Mirataei, M. Haidukevych, O. Markovskyi // Information, Computing and Intelligent systems. – 2022. – № 3.- P.56-62.
3. Гайдукевич М.А.. Метод безпечного розподіленого обчислення модулярної експоненти для прискорення реалізації механізмів захисту даних в IoT / М.А. Гайдукевич, Міратанаї Аліреза, О.В. Русанова // Проблеми управління та інформатизації.- 2023.- № 4 (77).- С.74-87. DOI: 10.18372/2073-4751.76.18243
4. Гайдукевич М. А.. Метод розподіленого модулярного експоненціювання на термінальних мікроконтролерах IoT із захищеним залученням хмарних обчислень / Русанова О.В., Гайдукевич М.А. // *Проблеми автоматизації та управління*, Т.2 № 78 (2024), С.91–103.

Ключові слова: модулярне експоненціювання; криптографія з відкритим ключем; захищеність систем IoT; цифровий підпис.

Пояснювальна записка

до магістерської дисертації

на тему: «Метод безпечного залучення хмарних систем для прискореної реалізації криптографії з відкритим ключем в Інтернеті Речей»

Київ – 2025

ЗМІСТ

ВСТУП

З моменту виникнення Інтернету наприкінці 1960-х років його можливості поступово еволюціонували від передачі інформації між комп'ютерами до повноцінної платформи для побудови розподілених систем керування та моніторингу стану. Одним із ключових напрямів [1] цього розвитку стало формування концепції Інтернету Речей (IoT) – мережі взаємопов'язаних пристроїв, здатних здійснювати обмін даними та керувати фізичними об'єктами в реальному часі. На сьогодні технології IoT знаходять широке застосування зокрема в галузях критичної інфраструктури, таких як сфери енергетики, медицини та логістики [2].

Застосування Інтернету як середовища передачі даних суттєво спрощує побудову та експлуатацію систем керування об'єктами реального світу. Це фактично дозволяє зменшити витрати на розгортання мереж, суттєво збільшує віддаленість пунктів керування і забезпечує високу адаптивність до змін у топології або функціоналі. Водночас використання Інтернету в якості середовища передачі інформації спричиняє ризики в контексті безпеки. Це пов'язано з тим, що мережа Інтернет вважається потенційно вразливим середовищем, оскільки існує потенційна можливість стороннього несанкціонованого втручання в процес інформаційного обміну. Зокрема такий ризик становить особливу загрозу для сфер критичної інфраструктури. Фактично мова йде про реалізацію таких процесів як здійснення моніторингу стану пацієнта в медичній інфраструктурі, керування безпілотними військовими літальними апаратами, віддалене управління транспортними засобами тощо. Для таких процесів виникає особлива потреба у забезпеченні надійного захисту інформаційного обміну, що здійснюється між пристроями систем управління [3].

Захист даних у системах IoT часто базується на механізмах цифрового підпису, таких як DSA або алгоритми, що відповідають стандарту ISO 979, у

яких ключову роль відіграє модулярне експоненціювання. Проте реалізація цієї операції на слабких за обчислювальними ресурсами термінальних пристроях викликає труднощі, оскільки довжина чисел, з якими здійснюються криптографічні перетворення, значно перевищує їхню розрядність [4]. Протягом останніх років стандартною довжиною чисел вважалась 2048, водночас останні рекомендації NIST встановлюють розрядність 4096 як ту, що відповідає сучасним вимоги до захисту інформаційних систем, тобто забезпечує надійний рівень захищеності секретних криптографічних компонентів. Разом з тим розглядається перспектива збільшення стандартної довжини компонентів криптографічних перетворень до 8192 протягом найближчих років. Очевидно, що вкрай обмежені можливості малих вбудованих термінальних мікроконтролерів на сьогоднішній день не дозволяють реалізувати такі складні операції в режимі реального часу. Тимчасом, саме такий режим необхідний для забезпечення захищеної взаємодії в системах віддаленого керування та моніторингу стану об’єктів реального світу. Зокрема, у сфері критичної інфраструктури прикладом можуть слугувати інтелектуальні транспортні системи, в яких IoT-пристрої відповідають за збір даних про трафік, координоване керування світлофорами, моніторинг транспортних засобів спеціального призначення тощо. У подібних системах затримки в обробці або передачі даних унаслідок недостатньої швидкодії криптографічних механізмів можуть призвести до збоїв у синхронізації світлофорних фаз, створення заторів або навіть до аварійних ситуацій.

Одним із перспективних шляхів вирішення цієї проблеми є залучення хмарних обчислювальних ресурсів для реалізації криптографічних операцій. Проте, у цьому випадку необхідно практичну неможливість, зокрема за рахунок економічної невигідності, відновлення справжній значень секретних операндів криптографії з відкритим ключем за даними, що передаються на хмарну систему. Зокрема це досягається шляхом впровадження гомоморфного шифрування, особливість якого дозволяє виконувати обчислення над

зашифрованими даними таким чином, щоб без їх розшифрування [5]. Розробка таких методів для безпечної реалізації модулярного експоненціювання в умовах хмарної обробки – актуальне завдання сучасної криптографії та важлива передумова для розвитку захищених систем IoT нового покоління.

РОЗДІЛ 1
АНАЛІТИЧНИЙ ОГЛЯД ІСНУЮЧИХ ПІДХОДІВ ДО ВИРІШЕННЯ ПРОБЛЕМИ ПРИСКОРЕННЯ ЗАХИЩЕНОЇ РЕАЛІЗАЦІЇ МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ

1.1 Аналіз задачі обчислення модулярної експоненти на термінальних пристроях IoT

У сучасних комп'ютерних системах та мережах ключовими напрямами захисту інформації є кілька важливих завдань. Насамперед, необхідно гарантувати відсутність стороннього доступу до даних, які передаються мережею або зберігаються на цифрових носіях, таких як жорсткі диски чи оптичні накопичувачі, що використовуються в комплексах для обробки інформації [6]. Це стосується як захисту від несанкціонованого читання, так і перегляду конфіденційних повідомлень.

Другою важливою метою є підтримка цілісності та достовірності інформації: потрібно запобігти будь-яким спробам змінити вміст повідомлень під час передачі по мережі або модифікувати збережені дані без належного дозволу. Йдеться про протидію спотворенню або фальсифікації інформації [7].

Також значну увагу приділяють обмеженню доступу до інформаційних систем та баз даних – ідентифікація й перевірка прав доступу користувачів є необхідним етапом при взаємодії з інтегрованими IT-середовищами [8]. Таким чином забезпечується, що лише авторизовані особи мають право працювати з конфіденційною інформацією.

Для ефективного вирішення завдань захисту інформації застосовується багатокомпонентна система, що поєднує в собі організаційні заходи, реалізовані, зокрема, через протоколи безпеки, спеціалізовані алгоритми, а також апаратні й програмні інструменти [9]. Ці елементи функціонують як

єдиний узгоджений механізм, спрямований на забезпечення надійного захисту даних на всіх етапах їх обробки та передачі. Особливу роль у цьому комплексі відіграють криптографічні методи, які широко впроваджуються для реалізації функцій конфіденційності, цілісності та автентифікації інформації.

Таке поняття як криптографія має низку переваг у порівнянні з виключно технічними методами захисту. Однією з головних її переваг є адаптивність: змінюючи лише ключ, можна оперативно налаштовувати рівень захисту відповідно до нових загроз або вимог. Крім того, стійкість криптографічних рішень меншою мірою залежить від розвитку апаратних засобів, які можуть бути використані зловмисниками для атак, що робить їх більш універсальними та перспективними в умовах постійної еволюції кіберзагроз.

Криптографія як основа інформаційної безпеки поділяється на два основні типи – симетричну та несиметричну. У симетричній криптографії один і той самий ключ використовується як для шифрування, так і для дешифрування даних. Прикладом таких алгоритмів є AES (Advanced Encryption Standard) [10], який забезпечує високу швидкість обробки даних і широко використовується в системах з обмеженими ресурсами.

Симетрична криптографія є одним з основних підходів до забезпечення конфіденційності та захищеності даних в сучасних інформаційних системах [11]. Вона базується на використанні одного ключа для як шифрування, так і для дешифрування інформації, що вимагає відправника та одержувача мати цей секретний ключ. Таким чином, секретність у системах, що використовують симетричну криптографію, забезпечується саме через приховування цього ключа від несанкціонованих осіб.

Основною перевагою симетричної криптографії є її швидкодія, оскільки операції шифрування та дешифрування зазвичай потребують менше часу і ресурсів у порівнянні з асиметричними методами, що робить її ідеальною для обробки великих обсягів даних. Відомі алгоритми симетричного шифрування, такі як AES (Advanced Encryption Standard), DES (Data Encryption Standard), а

також серія алгоритмів на основі блочного шифрування, забезпечують високий рівень безпеки та ефективності.

Однак, головним недоліком симетричної криптографії є проблема управління ключами. Оскільки для кожної пари комунікуючих осіб потрібен окремий секретний ключ, необхідно вирішувати задачу безпечної передачі цього ключа між сторонами, не розкриваючи його третіми особами. Зі збільшенням кількості користувачів в мережі, ця проблема стає дедалі складнішою, оскільки кожна пара користувачів повинна мати свій унікальний ключ, що збільшує складність управління ключами.

До того ж, для забезпечення безпеки в масштабах великих організацій, де необхідно постійно оновлювати ключі, проблеми з їхньою передачею і зберіганням стають ще більш актуальними [12]. Всі ці фактори ставлять під сумнів доцільність використання виключно симетричної криптографії в контексті великих або відкритих інформаційних систем, де безпечний обмін ключами і їхній захист вимагають великих ресурсів і постійної уваги.

У контексті управління віддаленими об’єктами, де необхідно забезпечити безпеку та автентифікацію на всіх етапах взаємодії між системами, симетричні алгоритми можуть стати недостатніми [13]. У таких системах, як правило, існує потреба в аутентифікації користувачів або пристроїв, що здійснюють доступ до віддалених об’єктів або передають важливі дані. Враховуючи високі вимоги до безпеки в таких сценаріях, ключовою проблемою є правильне управління доступом і забезпечення ідентифікації користувачів. Використання лише симетричних алгоритмів не дозволяє ефективно вирішити ці проблеми, оскільки для кожного новоутвореного з’єднання потрібно генерувати та обмінюватися секретними ключами, що стає важким завданням у разі великих, розподілених або динамічних мереж.

З цих причин, у таких системах, де важливе не лише шифрування даних, але й автентифікація учасників комунікації, значно ефективніше використовувати асиметричну криптографію [14]. Асиметричні методи, такі як

RSA, ECC (Elliptic Curve Cryptography) або DSA (Digital Signature Algorithm), використовують пару ключів: відкритий, що публічно розповсюджується, і закритий, який зберігається в секреті у користувача або пристрою. Цей підхід дозволяє вирішити проблему безпечної передачі ключа, оскільки відкритий ключ може використовуватися для шифрування або перевірки підпису, а закритий – для дешифрування або створення цифрового підпису.

Застосування асиметричної криптографії в системах керування віддаленими об’єктами дає змогу не тільки забезпечити конфіденційність переданих даних, але й верифікацію учасників комунікації [15]. Наприклад, за допомогою цифрових підписів можна підтвердити автентичність джерела інформації, а також забезпечити її цілісність, гарантувавши, що повідомлення не було змінено в процесі передачі. Така система значно покращує безпеку, оскільки кожен користувач або пристрій може перевірити автентичність іншого через публічний ключ, а доступ до закритого ключа залишається обмеженим.

Таким чином, хоча симетрична криптографія має свої переваги в контексті швидкості та ефективності, у складних і розподілених системах, де важлива не тільки конфіденційність, але й автентифікація та верифікація учасників, асиметрична криптографія є більш доцільним вибором. Це дозволяє забезпечити необхідний рівень безпеки, зберігаючи при цьому ефективність та масштабованість системи.

Натомість у несиметричній (асиметричній) криптографії використовуються дві пари ключів: відкритий (публічний) та закритий (приватний). Відкритий ключ застосовується для шифрування повідомлення, а приватний – для його дешифрування. Це дозволяє безпечно передавати інформацію навіть за умови відкритого каналу зв’язку.

Алгоритм RSA (Rivest–Shamir–Adleman) [16] є однією з найважливіших розробок у сфері несиметричної криптографії, що здійснила значний вплив на розвиток сучасних інформаційно-безпекових технологій. Його основна ідея полягає в тому, що для шифрування та розшифрування інформації

використовуються різні ключі: відкритий та закритий. Такий підхід дозволяє забезпечити безпечний обмін даними без необхідності попереднього передавання секретної інформації, як це має місце в симетричних методах шифрування.

Криптографічна стійкість RSA заснована на складності розв'язання задачі факторизації великих цілих чисел. Мова фактично йде про те, що маючи лише добуток двох великих простих чисел, реконструювати самі множники практично неможливо за сучасних обчислювальних потужностей. Саме ця математична особливість стає фундаментальним принципом для побудови криптографічних ключів.

Генерація ключів у RSA починається з вибору двох великих простих чисел p та q, які повинні бути різними та випадковими. Далі обчислюється добуток цих чисел $n = p \cdot q$, який використовується як частина відкритого та закритого ключа. Значення n також виступає модулем для подальших обчислень. Наступним кроком є визначення значення функції Ейлера $\varphi(n) = (p-1)(q-1)$, яке відображає кількість чисел, менших за n, що є з ним взаємно простими.

Після цього обирається відкрита експонента e, яка повинна бути взаємно простою з $\varphi(n)$, тобто не мати з нею спільних дільників, окрім одиниці. Визначення закритої експоненти d відбувається шляхом обчислення мультиплікативного оберненого до e за модулем $\varphi(n)$, тобто знаходження такого числа d, для якого виконується рівність $e \cdot d = 1 \bmod \varphi(n)$. У результаті формується пара криптографічних ключів: відкритий ключ e який може бути доступним будь-кому, та закритий ключ d, який повинен зберігатися конфіденційно.

Процедура шифрування в RSA здійснюється наступним чином. Початкове повідомлення M, яке повинно бути перетворене у числову форму меншу за n, шифрується за допомогою відкритого ключа за формулою $C = M^e \bmod n$, де C – це зашифрований текст. Дешифрування здійснюється з

використанням закритого ключа за формулою $M = C^d \bmod n$, що дозволяє отримати оригінальне повідомлення. Цей самий принцип, але у зворотному порядку (спочатку підпис – через d, перевірка – через e), використовується при створенні та перевірці електронного цифрового підпису.

До головних переваг алгоритму RSA відносять високий рівень безпеки за умови правильної реалізації, підтримку механізмів автентифікації, шифрування і цифрового підпису в одному рішенні, а також можливість відкритого розповсюдження ключа без шкоди для конфіденційності. Водночас варто зазначити, що RSA має певні обмеження: він є відносно повільним порівняно із симетричними алгоритмами, а обчислення, пов’язані з експоненціюванням великих чисел, потребують значних ресурсів, що ускладнює його використання на пристроях з обмеженою обчислювальною потужністю.

Незважаючи на це, RSA залишається фундаментальним алгоритмом у криптографії, часто використовується для безпечної передачі симетричних ключів, після чого шифрування великих обсягів даних відбувається швидшими методами, як-от AES. У сфері IoT, де пристрої часто мають обмежені ресурси, RSA може застосовуватись на етапі встановлення початкового захищеного з’єднання або для цифрової ідентифікації, проте у таких випадках часто виникає необхідність оптимізації обчислень або використання альтернативних алгоритмів, менш вимогливих до обчислювальних ресурсів.

Алгоритм Ель-Гамаля (ElGamal) [17] є ще одним фундаментальним алгоритмом несиметричної криптографії, який широко використовується реалізації цифрових підписів, шифрування даних, створення та забезпечення конфіденційності інформації. Він був запропонований у 1985 році Тахером Ель-Гамалем і ґрунтується на математичній складності задачі дискретного логарифмування в скінчених полях. На відміну від RSA, основна складність зламу якого пов’язана із факторизацією великих чисел, захищеність ElGamal забезпечується практичною складністю обчислення логарифма в мультиплікативній групі простого модуля.

У класичній формі ElGamal працює у скінченому полі Z^*_p, де p – велике просте число. Спочатку вибирається первісний корінь g, який є генератором мультиплікативної групи залишків за модулем p. Користувач генерує закритий ключ x, де $x \in [1, p\text{-}2]$, та обчислює відкритий ключ $y = g^x \bmod p$. Публічний ключ містить трійку параметрів (p, g, y), тоді як приватним ключем залишається лише значення x. Процедура шифрування повідомлення M, яке кодується у вигляді числа з інтервалу $[1, p\text{-}1]$, відбувається за допомогою випадкового тимчасового ключа k, який обирається на кожну сесію незалежно. Зашифрований текст у системі ElGamal є парою чисел (C_1, C_2), де $C_1 = g^k \bmod p$, а $C_2 = M \cdot y^k \bmod p$. Такий підхід називається стохастичним шифруванням, оскільки один і той самий відкритий ключ може утворювати різні зашифровані тексти для одного і того самого повідомлення за рахунок випадковості параметра k. Це суттєво підвищує криптографічну стійкість алгоритму.

Розшифрування повідомлення здійснюється за допомогою закритого ключа x. Спочатку обчислюється $s = C_1^x \bmod p$, що є спільним секретом, а потім повідомлення відновлюється за формулою $M = C_2 \cdot s^{-1} \bmod p$, в якому s^{-1} – це обернене до s число за модулем p. Як видно, лише власник закритого ключа може правильно обчислити s та розшифрувати передане повідомлення.

ElGamal має декілька важливих властивостей. По-перше, без знання закритого ключа навіть при перехопленні декількох зашифрованих текстів надзвичайно складно визначити вихідні повідомлення. По-друге, він забезпечує так звану семантичну безпеку, завдяки чому зловмисник не може зробити жодного припущення щодо зашифрованої інформації, навіть знаючи структуру можливих відкритих повідомлень. Однак, на відміну від RSA, ElGamal потребує подвоєного обсягу зашифрованого тексту (дві величини замість однієї), що робить його менш ефективним за обсягом при зберіганні та передаванні даних.

У рамках практичного застосування, алгоритм Ель-Гамаля активно використовується не лише для шифрування, але й для побудови цифрових підписів, зокрема в алгоритмі DSA (Digital Signature Algorithm) [18], який

базується на схожих математичних принципах. В умовах Інтернету Речей (IoT), де пристрої мають обмежені ресурси, використання ElGamal вимагає додаткової оптимізації, однак його криптографічні властивості, зокрема випадковість шифрування робить його придатним для захисту комунікацій між термінальними пристроями за умови коректної реалізації. У таких системах ElGamal найчастіше реалізується обчислювальними можливостями термінальних мікроконтролерів або ж з використання хмарних системи. Перед реалізаціє необхідно попередньою перевірити автентичність та захист каналу передачі даних.

Криптографія на еліптичних кривих (ECC, Elliptic Curve Cryptography) є одним із найсучасніших і найефективніших напрямів несиметричної криптографії [19], що забезпечує високий рівень захисту при відносно низьких обчислювальних витратах. Вона базується на алгебраїчних структурах, що виникають при операціях над точками, які належать до еліптичної кривої, визначеної над скінченим полем. Основною перевагою ECC є її здатність забезпечувати еквівалентний рівень криптографічної безпеки у порівнянні з традиційними методами (наприклад, RSA або ElGamal), але з використанням значно коротших ключів. Це особливо важливо в контексті обмежених обчислювальних ресурсів, зокрема в пристроях Інтернету Речей (IoT), де розмір пам’яті, енергоспоживання та пропускна здатність каналів передачі даних є критично важливими параметрами.

З точки зору математичного контексту еліптична крива над простим полем F_p, де p – велике просте число, описується рівнянням $y^2 = x^3 + ax + b$, де a і b – коефіцієнти, що визначають форму кривої, за умови, що дискримінант $4a^3 + 27b^2 \neq 0$. Елементи криптографічних схем ECC представляються у вигляді точок $P(x, y)$ на цій кривій. Основна криптографічна операція полягає в множенні точки P на скаляр k, тобто багаторазовому додаванні точки до самої себе: $Q = k \cdot P$. Ця операція є ефективною у прямому напрямку, але обернена задача – віднайдення k, знаючи лише P та Q, відома як задача дискретного

логарифма на еліптичній кривій (ECDLP) – вважається обчислювально складною навіть для найбільш потужних сучасних комп'ютерів.

Криптографічна стійкість ECC дозволяє використовувати ключі розміром 160–256 біт для досягнення безпеки, еквівалентної 1024–3072-бітним ключам у RSA. Наприклад, 256-бітний ключ ECC забезпечує рівень безпеки, співставний з 3072-бітним ключем RSA, при цьому значно зменшуючи час обчислень і використання пам'яті. Завдяки цим характеристикам ECC є оптимальним вибором для середовищ з обмеженими ресурсами.

До класичних протоколів на основі ECC належать алгоритм підпису ECDSA (Elliptic Curve Digital Signature Algorithm), алгоритм обміну ключами ECDH (Elliptic Curve Diffie–Hellman) та інші похідні схеми. Наприклад, при використанні ECDH для встановлення спільного секрету дві сторони обмінюються публічними точками $P_A = k_A \cdot G$ та $P_B = k_B \cdot G$, після чого обидві сторони незалежно обчислюють один і той самий спільний секрет $S = k_A \cdot P_B = k_B \cdot P_A$, що слугує основою для симетричного шифрування сесії.

Особливої актуальності ECC набуває в контексті IoT, де термінальні пристрої часто базуються на малопотужних мікроконтролерах і працюють у безперервному режимі. Саме тому економія енергії й швидкість виконання криптографічних операцій стають критичними. Реалізація ECC на таких пристроях дозволяє забезпечити автентифікацію, цілісність та конфіденційність даних без суттєвого навантаження на апаратні ресурси. Наприклад, ECC дозволяє здійснити обмін ключами або верифікацію цифрового підпису протягом кількох десятків мілісекунд, використовуючи кілька кілобайт оперативної пам'яті, що є досяжним навіть для 8-бітових мікроконтролерів.

Попри численні переваги криптографії на еліптичних кривих, зокрема високу ефективність та компактність ключів, що ідеально підходить для середовищ із обмеженими ресурсами, її поширення на практиці поки що залишається обмеженим [19]. Хоча ECC активно впроваджується в новітні

платформи безпеки та підтримується низкою сучасних протоколів, широке застосування цього підходу у вбудованих системах, особливо в екосистемі IoT, стикається з низкою об’єктивних технічних та інфраструктурних бар’єрів.

По-перше, не всі наявні архітектури термінальних мікроконтролерів, які використовуються в пристроях IoT, здатні ефективно реалізувати обчислення над еліптичними кривими. Часто йдеться про апаратні засоби, які були спроєктовані з урахуванням потреб традиційних симетричних алгоритмів або RSA, але не орієнтовані на виконання складних операцій над великими скінченними полями. Це обмежує можливість повноцінної апаратної реалізації алгоритмів на еліптичних кривих, зменшує швидкодію криптографічних функцій або ж потребує додаткових витрат на розробку оптимізованого програмного забезпечення для конкретної архітектури.

По-друге, відсутність єдиного стандарту щодо вибору конкретних кривих, параметрів та протоколів на основі ECC ускладнює їх впровадження в пристрої. У той час як деякі виробники мікроконтролерів інтегрують базову підтримку ECC (наприклад, через апаратне прискорення множення точок), інші обходяться лише програмною реалізацією або взагалі не передбачають використання ECC у типовому програмному стеку.

Отже, хоча криптографія на еліптичних кривих вбачається перспективною для використання у сфері інформаційної безпеки, зокрема в контексті IoT, її питання ефективної практичної реалізації цієї концепції залишається відкритим. Це обумовлено як обмеженнями апаратного рівня, так і відсутністю загальноприйнятих підходів до її інтеграції у архітектуру мікроконтролерів. З огляду на це, на поточному етапі розвитку ECC, попри її очевидні теоретичні та прикладні переваги, ще не набула загальноприйнятого стандарту у всіх класах IoT-пристроїв і потребує подальшого технологічного дослідження та адаптації.

Вагомі переваги зробили несиметричну криптографію важливою складовою безпеки у сфері Інтернету Речей (IoT), де мільярди пристроїв

передають чутливу інформацію у відкритих і потенційно вразливих середовищах. У таких умовах криптографічний захист є критично важливим, зокрема – для автентифікації пристроїв, шифрування переданих даних та забезпечення їх цілісності.

В системах Інтернету Речей з'являється особливий виклик: обмежені ресурси термінальних пристроїв, таких як сенсори або виконавчі модулі, які мають мінімальні обчислювальні потужності та енергоспоживання [20]. Реалізація криптографічних механізмів у таких умовах покладається на термінальні мікроконтролери – спеціалізовані мікросхеми, які виконують базові криптографічні операції без залучення потужних серверів.

Таким чином, саме термінальні пристрої стають ключовим елементом у забезпеченні захисту даних в IoT-мережах, реалізуючи алгоритми шифрування, підпису, автентифікації та перевірки цілісності без надмірного навантаження на мережеву інфраструктуру.

Швидка реалізація операції модулярного експоненціювання $D^K \bmod M$, яка є основою криптографії з відкритим ключем, постає критично важливим питанням для малопотужних термінальних платформ, що використовуються в системах моніторингу та управління об'єктами реального світу на базі технологій IoT [21].

Однією з головних проблем при використанні термінальних мікроконтролерів для обчислень у системах IoT є обмеження енергоспоживання [22]. Багато таких пристроїв працюють від батарейок або інших обмежених джерел енергії, що робить критичним питання оптимізації споживаної потужності. У цьому контексті важливо не тільки забезпечити швидкість виконання обчислень, а й мінімізувати енергетичні витрати при передачі та обробці даних.

Враховуючи різноманітність архітектур мікроконтролерів, що використовуються в IoT-системах (наприклад, ARM, x86, RISC-V), важливо адаптувати методи модулярного експоненціювання під конкретні платформи.

Кожна архітектура має свої переваги і обмеження, тому для досягнення найкращої продуктивності необхідно використовувати спеціалізовані інструкції та апаратні можливості, доступні на кожній конкретній платформі.

Наприклад, на платформі ARM можна використовувати SIMD (Single Instruction, Multiple Data) інструкції для паралельної обробки кількох даних одночасно, що дозволяє значно прискорити операції, пов'язані з експоненціюванням. Крім того, деякі мікроконтролери мають спеціалізовані криптографічні прискорювачі [23], що можуть бути використані для швидкого виконання операцій, таких як модулярне експоненціювання.

Також важливо враховувати ресурси, доступні на конкретному мікроконтролері, зокрема обсяг пам'яті та швидкість виконання інструкцій. Для деяких пристроїв необхідно розробляти спеціалізовані алгоритми, які дозволяють зменшити обсяг пам'яті, що використовується для зберігання проміжних результатів, а також забезпечити ефективну обробку даних, щоб уникнути перевантаження ресурсів пристрою.

На сучасному етапі розвитку інформаційно-керованих систем, зокрема в контексті Інтернету Речей (IoT), гостро постає питання забезпечення надійного криптографічного захисту. Відповідно до рекомендацій Національного інституту стандартів і технологій США (NIST), для забезпечення належного рівня стійкості до криптоаналітичних атак у довгостроковій перспективі, стандартний розмір криптографічного ключа для алгоритмів на основі модулярного експоненціювання (таких як RSA чи El-Gamal) повинен становити не менше ніж 4096 бітів [24]. Такі параметри дозволяють протистояти сучасним обчислювальним потужностям потенційного зловмисника та відповідають вимогам до високозахищених інформаційних систем.

Проте в умовах реального використання термінальних вбудованих пристроїв, таких як мікроконтролери, які функціонують у системах дистанційного моніторингу й управління об'єктами реального світу, реалізація модулярного експоненціювання з ключами подібної довжини вбачається

надзвичайно ресурсоємним завданням. Обмежена тактова частота, невеликий об’єм оперативної пам’яті, відсутність апаратної підтримки великих цілих чисел – усе це унеможливлює виконання повноцінних криптографічних операцій у режимі реального часу, що є обов’язковою вимогою для ефективного функціонування систем віддаленого управління.

Одним із найбільш перспективних підходів до вирішення цієї проблеми є комбінування локальних і хмарних обчислень для зменшення енергоспоживання [25]. Частину обчислень можна здійснювати безпосередньо на пристрої, а складніші або ресурсоємні операції передавати в хмару. Такий підхід дозволяє знизити навантаження на локальні мікроконтролери та зменшити потребу в постійних високих обчислювальних потужностях. Крім того, важливо враховувати час, коли пристрій працює на високих навантаженнях, і коли його можна перевести в режим енергозбереження, тим самим оптимізуючи використання енергії.

Також можна використовувати техніки, які дозволяють пристрою змінювати свої робочі режими в залежності від потреб. Наприклад, мікроконтролери можуть переходити в сплячий режим, коли не відбуваються обчислення або передача даних, а активуватися лише в моменти, коли це необхідно. Такий підхід може значно знизити витрати енергії і продовжити термін служби батареї.

IoT-системи часто працюють в умовах, де мережа може бути нестабільною або мати обмежену пропускну здатність. В таких випадках важливо, щоб методи обчислень могли адаптуватися до зміни умов мережі в реальному часі. Наприклад, коли мережа є швидкою і стабільною, обчислення можуть частково або повністю передаватися в хмару для виконання більш складних операцій. Однак, коли мережа має низьку пропускну здатність або є переривчастою, необхідно використовувати стратегії, які дозволяють зберігати більшу частину обчислень на локальному пристрої.

Для цього можна використовувати динамічне управління обчисленнями, яке постійно відстежує стан мережі і визначає оптимальний розподіл обчислювальних задач між пристроєм і хмарою. Така адаптивність дозволяє мінімізувати затримки і підвищити ефективність роботи системи. Наприклад, при низькому рівні сигналу або нестабільному з'єднанні, можна вибрати стратегію виконання операцій на пристрої, зменшуючи необхідність в передачі даних по мережі.

Крім того, при реалізації таких стратегій важливо враховувати час затримки між пристроєм і хмарою, який може суттєво змінюватися в залежності від навантаження на мережу. Інтелектуальна система може визначити, які обчислення повинні бути виконані в хмарі, а які можна обробити локально, таким чином покращуючи загальну ефективність роботи пристроїв. Інтеграція блокчейн-технологій [26] в системи IoT може істотно підвищити безпеку та надійність обміну даними. Блокчейн дозволяє зберігати інформацію в розподіленій базі даних, що робить її незмінною та прозорою для всіх учасників мережі. Для IoT це є важливою перевагою, оскільки вона дозволяє захистити дані від несанкціонованих змін і атак.

Відповідно до цього, постійно з'являються нові методи, що застосовують хмарні обчислення для підвищення ефективності цих операцій.

1.2 Огляд новітніх підходів вирішення задачі прискорення захищеного модулярного експоненціювання на термінальних пристроях IoT

В теоретичному плані, існуючі підходи можна розглядати як варіанти спеціалізованого гомоморфного шифрування [27], де процес обчислення $D^K \bmod M$ здійснюється віддалено, з розподілом операцій між термінальними пристроями та хмарними сервісами.

Ефективність таких методів оцінюється за кількома критеріями: перший – це прискорення виконання модулярного експоненціювання на термінальному

мікроконтролері за рахунок використання віддалених обчислень, а другий – рівень безпеки, який забезпечується захистом від потенційних атак, спрямованих на відновлення секретних компонентів операції через дані, що передаються в хмару. Процес обробки секретного коду експоненти умовно поділяється на дві частини: одну частину обробляє мікроконтролер, а іншу – хмара. Важливою є пропорція між цими частинами, оскільки вона визначає, наскільки ефективно працює метод: чим більше розрядів обробляється на мікроконтролері, тим вищий рівень захисту, однак це також може впливати на швидкість обчислень.

Залучення криптопроцесорів для виконання операцій модулярного експоненціювання в хмарі дозволяє значно підвищити швидкість обчислень, оскільки такі пристрої здатні значно прискорити виконання складних криптографічних операцій [28]. Однак, використання криптопроцесорів також має суттєвий недолік: неможливість передавати проміжні результати на термінальний мікроконтролер, що ускладнює організацію ефективних комбінованих обчислень між термінальними і хмарними платформами. У результаті, існуючі методи можна поділити на дві категорії: перша включає методи, що використовують криптопроцесори, а друга – ті, що базуються на потужних процесорних засобах віддалених систем без використання криптопроцесорів.

Методи першої категорії характеризуються високою швидкістю обчислень, але через неможливість передачі проміжних результатів на мікроконтролер вони вимагають поділу процесу на кілька етапів. Спочатку відбувається виконання попередніх обчислень на термінальному мікроконтролері, потім – часткове модулярне експоненціювання в хмарі, а остаточний результат повертається на мікроконтролер. Операції обчислення засновані на мультиплікативно-адитивному розкладі коду експоненти $K = U \cdot F + W$ [29], де значення U шифрує інформаційну складову D, тим самим дозволяючи уникнути її передачі в явному вигляді в хмару. Змінна U, що

характеризується малою бітовою довжиною, використовується для шифрування інформаційної компоненти D. Це дозволяє уникнути надсилання D у відкритому вигляді на віддалену обчислювальну платформу для виконання часткового експоненціювання по модулю. Значення W при цьому відіграє роль у виборі оптимального значення U, забезпечуючи ефективність загального процесу. Схема обчислення виразу $D^K \bmod M$ передбачає декілька етапів. Спершу, безпосередньо на термінальному пристрої виконуються дві операції: обчислення $Y = D^U \bmod M$ і $S = D^W \bmod M$. Результат Y разом з частиною експоненти F та відкритим модулем M передається на зовнішній сервер. Далі у хмарному середовищі виконується обчислення $Q = Y^F \bmod M$, після чого отримане значення Q повертається назад на термінал. Заключний етап полягає у виконанні операції модулярного множення між Q і попередньо знайденим S, що дає кінцевий результат: $R = (Q \cdot S) \bmod M$. Існують також інші відомі підходи [30,31] до пришвидшеного обчислення модулярної експоненти на мікроконтролерах, які базуються на використанні більш складних форм представлення експоненти у вигляді комбінованих мультиплікативно-адитивних кодів.

Застосування підходів, які належать до другої групи методів, забезпечує значно вищу ефективність розподілу обчислювального навантаження між термінальним мікроконтролером та хмарними ресурсами, що досягається завдяки розширеним можливостям обміну даними. У таких підходах код експоненти часто розкладається на дві компоненти, що дозволяє здійснювати паралельні обчислення, а отримані проміжні результати регулярно передаються назад на термінальний пристрій для подальших розрахунків. Залучення таких методів дозволяє збільшити кількість розрядів, які обробляються безпосередньо на термінальній платформі, тим самим підвищуючи рівень захисту, оскільки зловмиснику необхідно буде відновлювати не тільки самі розряди, але й їх позиції в коді експоненти.

Типовим прикладом реалізації швидкого модулярного експоненціювання із залученням хмарної інфраструктури є метод, описаний у роботі [32]. У цьому методі реалізовано безпечний обмін проміжними результатами між хмарою та термінальною платформою, які надалі використовуються для паралельних обчислень на мікроконтролері. В основі методу лежить подання експоненти у вигляді $K = (Y + X) \cdot 2 + \beta$, де β є залишком від ділення експоненти на два.

Особливістю зазначеного підходу є специфіка розподілу обчислень між обома платформами. На початковому етапі термінальна система виконує модулярне піднесення числа D до степеня β, формуючи значення $S = D$ в степені β за модулем M. Паралельно з цим здійснюється шифрування D через модулярне піднесення до квадрату – $L = D$ в квадраті за модулем M. Далі на хмарний сервер передаються отримане значення L, частковий код експоненти Y та модуль M. У хмарі на основі цих параметрів обчислюється проміжний результат $P = L$ в степені Y за модулем M. Сам процес розбитий на певну кількість інтервалів, наприкінці кожного з яких проміжні значення повертаються на термінальний пристрій. Паралельно на мікроконтролері виконується обчислення $Q = B$ в степені X за модулем M на основі отриманих з хмари даних. Остаточне значення R формується у вигляді $R = S$ помножити на Q і на P за модулем M. Такий підхід дозволяє збільшити кількість бітів експоненти, оброблюваних безпосередньо на мікроконтролері, що підвищує рівень захищеності системи. До того ж, для несанкціонованого відновлення повного коду експоненти зловмиснику потрібно не лише перебрати усі можливі значення оброблюваних бітів, а й визначити їхнє розташування у повному коді.

Разом з тим, методи другої групи мають спільний недолік, що полягає у невисокій продуктивності хмарної частини обчислень. Це обумовлено відсутністю використання можливостей сучасних апаратних криптоприскорювачів, здатних значно підвищити швидкість виконання відповідних операцій.

Інші методи, що включають комбіновану організацію обчислень між хмарними та термінальними платформами, також дозволяють зберегти високу швидкість обчислень, одночасно збільшуючи кількість розрядів, які обробляються на термінальному пристрої. Це дає змогу підвищити рівень захисту без суттєвого зниження швидкодії системи. Проте, основним обмеженням таких підходів є відсутність використання криптопроцесорів, що знижує їх загальну продуктивність порівняно з методами першої категорії.

Для покращення захищеності операндів D і K при модулярному експоненціюванні $D^K \bmod M$ в контексті віддаленого виконання запропоновано метод комбінованого використання адитивних ланцюжків, передобчислень і маскування [33]. У цьому методі секретний код експоненти K, який має вигляд $K = k_0 + 2 \cdot k_1 + 2^2 \cdot k_2 + \ldots + 2^{n-1} \cdot k_{n-1}$, де кожен елемент k_i належить множині $\{0, 1\}$, розбивається на фрагменти довжиною m, тобто таких фрагментів n/m. Кожен фрагмент f_k складається з m бітів, наприклад: $f_0 = \{a_0, a_1, \ldots, a_{m-1}\}$, $f_1 = \{a_m, a_{m+1}, \ldots, a_{2m-1}\}$ і так далі. Модулярне експоненціювання пропонується здійснити в два етапи:

1. Передобчислення значень $A^2 \bmod M$, $A^3 \bmod M$, …, $A^{\varphi} \bmod M$, де $q = 2^m - 1$, і збереження цих результатів у таблиці T, де $T[0] = 1$, $T[1] = A$, $T[2] = A^2 \bmod M$, …, $T[q] = A^{\varphi} \bmod M$.

2. Безпосереднє модулярне експоненціювання з використанням передобчислених значень виконується в кілька кроків:

- Спочатку призначається значення k для фрагмента експоненти, який містить старші біти A: $k = n/m$. Змінній R встановлюється значення 1: $R = 1$.

- Потім змінній R присвоюється результат модулярного добутку попереднього значення R на d_k – значення з таблиці передобчислень, яке відповідає поточному фрагменту: $R = R \cdot T(d_k) \bmod M$. Значення d_k формується за допомогою бітів поточного фрагмента f_k.

• Наступним кроком є обчислення модулярного квадрата R і інкремент змінної i: $i = i + 1$. Якщо $i < m$, то процес повторюється.

• Після обробки всіх фрагментів, включаючи останній, виконується модулярне множення значення R на таблицю передобчислень для наймолодшого фрагмента.

У результаті отримується код $R = D^K \bmod M$.

Процес піднесення до квадрату та модулярного добутку в разі великої розрядності чисел (наприклад, 1024 або 2048) реалізується через множення секцій співмножників. Час обробки кожного фрагмента для m розрядів визначається як час m операцій модулярного піднесення до квадрату і одного модулярного добутку. Для останнього фрагмента час виконання обчислення складається лише з одного модулярного добутку. Загальний час безпосереднього обчислення модулярної експоненти може бути визначений як: $T_1 = (n/m - 1) \cdot m \cdot t_q + n/m \cdot t_{mult}$, де t_q – час модулярного піднесення до квадрату, t_{mult} – час модулярного добутку.

При роботі з великими числами множення і піднесення до квадрату реалізуються через розбиття чисел на секції, що зменшує кількість операцій. Якщо секцій $s = 32$, то кількість множень при модулярному піднесенні до квадрату зменшується до $(s - 1) \cdot s / 2$. Модифікована формула для оптимізації часу обчислення виглядає так: $T_1 = ((2^{m-1} - 1) \cdot 3/2 + (n - m) \cdot 1/2 + n/m) \cdot t_{mult} = \chi \cdot t_{mult}$, де χ – коефіцієнт, який визначає кількість модулярних добутків. Експериментальні дані показують [33], що оптимальне значення m для будь-якої розрядності n становить близько 8.

Таким чином, огляд існуючих методів швидкого обчислення модулярної експоненти для малопотужних термінальних мікроконтролерів, що використовують хмарні технології, показує, що жоден з них повністю не відповідає вимогам до ефективності та безпеки, які необхідні для забезпечення криптографічних механізмів в сучасних системах моніторингу та управління IoT.

Висновки до розділу 1

В результаті проведених досліджень, які складають перший розділ магістерської дисертації і спрямовані на огляд існуючих підходів вирішення проблеми прискорення захищеної реалізації модулярного експоненціювання та пошук найбільш перспективних шляхів підвищення ефективності, можуть бути зроблені наступні висновки:

1. Проведений аналіз практичного використання криптографії з відкритим ключем показав, що найбільш критичним вбачається застосування цих технологій в системах віддаленого моніторингу та керування об'єктами реального світу. При цьому основним чинником виступає обмеженість обчислювальних ресурсів термінальних комп'ютерних платформ, які часто не можуть забезпечити реалізацію функцій контролю автентичності даних та команд в режимі реального часу. З огляду на те, що термінальні мікроконтролери оснащені радіомодемом, одним з найбільш перспективних шляхів підвищення ефективності реалізації криптографічних протоколів з відкритим ключем на цих пристроях полягає у залученні до обчислень віддалених комп'ютерних систем.
2. В результаті аналізу відомих технологій залучення хмарних систем виявлено два можливих варіанти організації розподілених обчислень модулярної експоненти. Перший з них передбачає наявність на віддаленій платформі спеціалізованого криптопроцесора, обчислювальна потужність якого дозволяє реалізувати модулярне експоненціювання на порядки швидше, ніж на термінальному мікроконтролері. Водночас, застосування криптопроцесора на віддаленій системі не дозволяє обмінюватись проміжними результатами обчислень з мікроконтролером. Другий спосіб організації розподілених обчислень полягає у використанні в хмарі високопотужних процесорних засобів без залучення криптопроцесорів. Такий варіант організації передбачає можливість обміну проміжними результатами обчислень з мікроконтролером. Проведені дослідження показали, що більш доцільним є

реалізація саме другого способу розподілу обчислень модулярної експонети.

3. Детальний аналіз алгоритмів модулярного експоненціювання показав, що більші перспективи для розподіленого обчислення надає алгоритм експоненціювання з молодших розрядів, який, фактично, містить дві гілки обчислень, одна з яких може вважатися несекретною: а саме n операцій послідовного піднесення до квадрату числа, над яким виконується експоненціювання. Ця гілка може повністю виконуватися на віддаленій комп'ютерній платформі. Обчислення другої гілки частково або повністю можуть бути реалізовані на термінальній обчислювальній платформі з урахуванням результатів віддалених обчислень. Потрібно дослідити ефективні форми такої взаємодії для розробки ефективних методів організації розподілених обчислень.

РОЗДІЛ 2.РОЗРОБКА МЕТОДУ РОЗПОДІЛЕНОГО МОДУЛЯРНОГО ЕКСПОНЕНЦІЮВАННЯ НА ТЕРМІНАЛЬНИХ МІКРОКОНТРОЛЕРАХ ІЗ ЗАХИЩЕНИМ ЗАЛУЧЕННЯМ ХМАРНИХ СИСТЕМ

2.1 Дослідження можливих способів організації розподіленого модулярної експоненти на термінальних мікроконтролерах

Операція модулярного експоненціювання $D^K \bmod M$ в запропонованому методі базується на використанні одного з двох класичних алгоритмів модулярного експоненціювання. Обидва алгоритми полягають у виконанні n-ітераційного циклу, причому, на кожній ітерації здійснюються операції, які залежать від значення поточного двійкового розряду коду експоненти [34]. Відповідно, згадані вище алгоритми відрізняються порядком, в якому здійснюється обробка коду експоненти: перший аналізує код експоненти починаючи зі старших розрядів, натомість другий передбачає модулярне експоненціювання починаючи з молодших розрядів коду експоненти. Алгоритм зі старших розрядів полягає у реалізації n-ітераційного циклу модифікацій змінної R, початкове значення якої дорівнює одиниці. Під час кожної ітерації передбачається послідовне виконання операцій модулярного піднесення числа R до квадрату та модулярного множення R на постійне число A. При цьому обчислення останньої операції – модулярного добутку – здійснюється на тих ітераціях циклу, на яких значення поточного розряду двійкового коду експоненти дорівнює одиниці. Кінцевим результатом обчислень є число R, отримане на останній ітерації циклу. Алгоритм такого типу є строго послідовним, тобто під час його реалізації здійснюється модифікація однієї змінної. Це не дозволяє організувати його розпаралелення при програмній реалізації. Відповідно залучення кількох паралельних обчислювальних процесів на віддаленій хмарній системі не є доцільним. Суть алгоритму модулярного експоненціювання з молодших розрядів полягає у виконанні n ітерацій циклу модифікацій двох змінних A та R, значення яких на початку

алгоритму дорівнюють A та 1 відповідно. На кожній ітерації передбачено виконання двох операцій: модулярного множення $A \cdot R \bmod M$, із збереженням результату у змінну R, та модулярного піднесення до квадрату $A = A^2 \bmod M$, із збереженням результату у змінну A. Модифікація значення R здійснюється у випадку, коли поточний біт коду експоненти рівний одиниці, в той час як обчислення модулярного піднесення до квадрату реалізується на кожній ітерації циклу незалежно від значення поточного розряду коду експоненти. Кінцевим результатом модулярного експоненціювання є значення модулярного добутку, отримане на останній ітерації циклу та записане у змінну R. Таким чином характер алгоритму з молодших розрядів дозволяє організувати два паралельні обчислювальні процеси для обрахунку двох компонент: A та R. Тому для досягнення значного рівня прискорення в обчисленні модулярної експоненти з залученням двох паралельних віддалених процесів є доцільним використання алгоритму модулярного експоненціювання з молодших розрядів [35].

Один із перспективних напрямів оптимізації модулярної експоненти на IoT-пристроях полягає в реалізації частково делегованої обчислювальної моделі, що передбачає суміщення локального та віддаленого виконання [36]. Суть підходу полягає в організації двох обчислювальних процесів, фізично або логічно відокремлених, з чітким розподілом функцій між ними. Перший процес виконується безпосередньо на термінальному мікроконтролері й відповідає за всі операції, які не передбачають обробку секретної частини експоненти або інші чутливі обчислення. Це можуть бути, зокрема, попередні трансформації повідомлення, генерація та зберігання публічних параметрів, підготовка частин основи до піднесення в степінь, а також локальне управління комунікацією з обчислювальним бекендом [37]. Другий процес, що виконується у віддаленій системі (наприклад, у хмарній інфраструктурі), здійснює обчислення окремих компонентів модулярної експоненти. При цьому використовуються криптографічні методи розщеплення експоненти (наприклад, адитивного

розкладення), які дозволяють забезпечити обчислення без доступу до повної експоненти, що гарантує конфіденційність секретного ключа навіть у разі компрометації хмарної компоненти [38]. У межах такого підходу можливе використання методів:

- Розподіленого експоненціювання [39] з адитивним розкладенням експоненти: експонента поділяється на дві або більше частини (наприклад, $K = K_1 + K_2$), де K_1 використовується локально, а K_2 обчислюється у хмарі. Остаточний результат формується через комбінування часткових результатів.
- Рандомізації обчислень: вводяться псевдовипадкові обертання або маскування базових параметрів, які знімають кореляцію між вхідними даними й вихідними результатами, запобігаючи побічним атакам.
- Безпечного об’єднання результатів: за допомогою попередньо домовлених криптографічних схем, що дає змогу звести часткові обчислення у фінальне значення $D^K \bmod M$ без розкриття жодної з частин повної експоненти.

Такий підхід не лише дозволяє суттєво зменшити навантаження на мікроконтролер, але й відкриває можливість адаптивного масштабування криптографічних обчислень відповідно до змін у доступних ресурсах та мережевій інфраструктурі. Окрім цього, він створює додатковий рівень захисту від атак побічних каналів, оскільки критичні частини обчислень виконуються не в середовищі, де знаходиться фізичний пристрій [40]. Модулярне експоненціювання виду $D^K \bmod M$ включає три основні числові елементи: самі дані D, секретний ключ K, і модуль M, який є відкритим параметром. Обробка K на зовнішніх, не контрольованих пристроях є критично чутливою з точки зору безпеки, на відміну від змінного повідомлення D. Причина полягає в тривалому та багаторазовому використанні закритого ключа, яке при компрометації дозволяє стороннім особам здійснювати несанкціоновані операції, тоді як D постійно змінюється і нерідко є загальнодоступним, наприклад, у системах цифрового підпису на мікроконтролерах. Проведений аналіз показав, що рівень захищеності обчислень модулярної експоненти з

використанням можливостей віддалених обчислювальних процесів визначається кількістю розрядів секретного коду експоненти, обробка яких здійснюється на термінальному мікроконтролері. Тому для підвищення рівня захищеності доцільна обробка мікроконтролером якомога більшої кількості розрядів коду експоненти. Це може бути досягнуто за рахунок делегування одному з віддалених процесів виконання всіх об'ємних допоміжних обчислень, з одночасною реалізацією на термінальному мікроконтролері проведення лише тих обчислень, які безпосередньо залежать від секретного коду експоненти.

Показники ефективності використання хмарної системи при реалізації експоненціювання визначаються двома чинниками: скороченням часу обчислень на мікроконтролері завдяки залученню потужностей віддалених систем, а також здатністю убезпечити приватний ключ від відновлення через передані у хмару дані. Максимальна ефективність досягається шляхом одночасного виконання обчислень як на мікроконтролері, так і на віддаленому сервері, а також завдяки реалізації паралельної обробки на стороні хмари. Аналіз показав, що ступінь безпеки залежить від кількості бітів експоненти, які залишаються на стороні клієнта. Чим більше таких бітів обробляється локально, тим вищий захист. Однак це обмежується ресурсами термінального пристрою. Щоб підвищити рівень захищеності, не втрачаючи продуктивність, доцільно знизити обчислювальне навантаження на локальний пристрій. Це можна реалізувати через поділ процесу на дві частини: одну – для обробки чутливих розрядів – залишити локально, іншу – перенести в хмару. Серед поширених підходів – поділ обчислень за ітераціями: частина з n ітерацій виконується на мікроконтролері, решта – в хмарі. Зазвичай x молодших бітів обробляються локально, тоді як n-x старших – віддалено. Значення x обирають для досягнення максимальної синхронності між локальними й віддаленими процесами, а також відповідного рівня безпеки. Якщо позначити g – кількість бітів, які не повинні бути передані в хмару відповідно до вимог захисту, тоді $x \geq g$. У разі, якщо обчислення g бітів займає менше часу, ніж обробка n-g на сервері, x можна

збільшити для досягнення рівноваги за часом. Інакше, досягнути одночасності не вдається. Традиційні методи передбачають, що кожен біт коду експоненти обробляється локально через виконання двох операцій в кожній ітерації – як множення, так і піднесення до квадрату. Один з можливих підходів розширює цей розподіл: операції розділяються не лише по ітераціях, а й всередині кожної з них. Зокрема, при використанні алгоритму, у якому обробка починається з молодших бітів, де в кожній ітерації виконуються дві незалежні дії: $R \cdot A \bmod M$ і $A^2 \bmod M$. Перша операція залежить від значення біта експоненти (виконується, якщо біт дорівнює 1, отже, вона має залишатися локальною. Натомість обчислення $A^2 \bmod M$, не залежне від поточного біту коду, може бути делеговане хмарі.

У повному варіанті реалізації поділу на рівні операцій, секретна частина експоненти повністю обробляється на клієнтському пристрої. Таким чином, виникають два паралельні потоки: перший – локальний, виконує послідовність модулярних множень, використовуючи значення біта експоненти; другий – хмарний, обробляє незалежні піднесення до квадрату. Для кожної i-тої ітерації, $i \in \{1,2,\ldots,n\text{-}1\}$, результат $A_{i\text{-}1}$ з другого процесу необхідний для виконання $R \cdot A \bmod M$ на мікроконтролері.

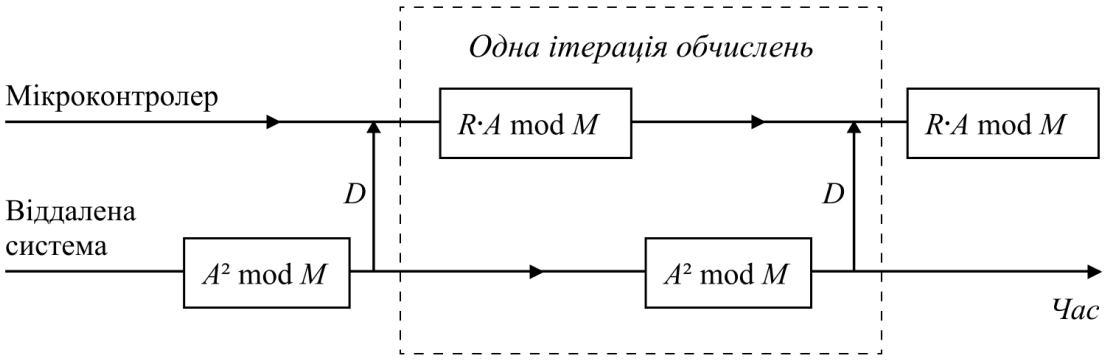

Рис. 2.1. Можлива організація обчислювального процесу у зазначеному підході

Слід підкреслити, що обчислювальні ресурси хмарної інфраструктури суттєво переважають можливості термінального мікроконтролера. Саме тому виконання серії обчислень типу $A^2 \bmod M$ на віддаленому сервері потребує в рази менше часу, ніж формування відповідного набору результатів R на локальному пристрої. Це зумовлено не лише різницею в тактовій частоті або кількості доступних ядер процесора, а й наявністю оптимізованих алгоритмів та підтримкою паралельного обчислення в хмарному середовищі.

Завдяки тому, що обидва процеси – обчислення послідовностей A та R – виконуються незалежно один від одного в рамках кожної ітерації алгоритму експоненціювання, відпадає необхідність у строгій синхронізації між локальним та хмарним виконанням. Іншими словами, кожен з потоків може виконуватися асинхронно, відповідно до власних обчислювальних можливостей, не очікуючи завершення ітерації на іншій стороні. Варто також враховувати специфіку передавання даних через мережу Інтернет, яка використовується як транспортний канал між клієнтом і сервером. Через затрати часу на встановлення з’єднання, обробку протоколів і передачу кожного окремого пакету, надсилання одного результату A після кожної ітерації є надзвичайно неефективним. Формування мережевого пакета часто триває довше, ніж саме модулярне піднесення до квадрату, яке він має передати. Тому з погляду оптимізації доцільно застосувати підхід попереднього обчислення і акумуляції множини значень A на віддаленому сервері з подальшою одноразовою передачею цього пакета на клієнтський мікроконтролер. Такий підхід дозволяє мінімізувати часові затримки, зменшити кількість мережевих звернень та зберегти обчислювальну перевагу хмарної платформи. У підсумку, він забезпечує ефективний розподіл навантаження між клієнтським та серверним середовищем із урахуванням як архітектурних особливостей, так і характеристик мережевого середовища.

2.2 Розробка методу розподіленого модулярного експоненціювання на термінальних мікроконтролерах IoT з захищеним залученням хмарних систем

Фундаментальна ідея, покладена в основу розробленого методу, полягає у альтернативній організації розподілених обчислень між хмарною системою та термінальним мікроконтролером. Для забезпечення конфіденційності інформаційно' компоненти D в процесі віддаленого виконання модулярного експоненціювання $D^K \bmod M$, у запропонованому підході передбачено попереднє шифрування цієї компоненти. Шифрування реалізується безпосередньо на термінальному мікроконтролері шляхом виконання операції модулярного піднесення до квадрату: $B = D^2 \bmod M$. Паралельно з цим виконується поділ початкового n-розрядного експонентного коду K на два, що еквівалентно зсуву вправо на один біт: $S = K >> 1$. Отриманий результат S має n-1 розряд і використовується для подальшого обчислення експоненти $B^S \bmod M$ вже у віддаленому середовищі. Перед початком основного обчислення на мікроконтролері змінна R ініціалізується залежно від парності коду K: якщо K – парне, то $R = 1$, інакше $R = D$. Це відповідає операції $R = D^K \bmod 2$. Код S, що має вигляд $S = s_1 \cdot 2^0 + s_2 \cdot 2^1 + \ldots + s_{n\text{-}1} \cdot 2^{n\text{-}2}$, де $\forall\, l \in \{1,2,\ldots, n\text{-}1\}: p_l \in \{0,1\}$ додатково поділяється на j сегментів. У кожному з цих сегментів частина бітів (a розрядів) залишається прихованою (тобто не передається), а решта (b розрядів) передається для обробки у хмарну систему. Таким чином, загальна довжина кожного сегмента становить $a + b$ бітів. У результаті такого поділу і трансформації формується новий код $Q = q_1 \cdot 2^0 + q_2 \cdot 2^1 + \ldots + q_{n\text{-}1} \cdot 2^{n\text{-}2}$, який, разом із величинами B і M, передається на віддалений сервер для виконання відповідних обчислень. Код Q є результатом наступної модифікації коду S:

$$\forall\, l \in \{1,2,\ldots,n-1\}: \begin{cases} l \bmod (a+b) \le b: q_l = s_l; \\ l \bmod (a+b) > b: q_l = 0. \end{cases} \quad (2.1)$$

В хмарній системі змінній A_0 присвоюється значення B: $A_0 = B$. Першим кроком є послідовна обробка всіх j сегментів трансформованого коду експоненти Q у такому порядку.

На i-тому сегменті, $i \in \{1, j\}$, реалізується рекурентне обчислення величин $A_{(i\text{-}1)(a+b)+1} = A_{(i\text{-}1)(a+b)}^2 \bmod M$, $A_{(i\text{-}1)(a+b)+2} = A_{(i\text{-}1)(a+b)+1}^2 \bmod M$, …, $A_{i(a+b)} = A_{i(a+b)\text{-}1}^2 \bmod M$. Окрім цього, на основі молодших b розрядів сегменту експоненти Q, здійснюється обчислення значення W_i:

$$W_i = \left(\prod_{k=1}^{b} A_{k+(i-1)(a+b)-1}^{\,q_{k+(i-1)(a+b)}}\right) \bmod M \quad . \qquad (2.2)$$

Наступний етап запропонованого методу передбачає обчислення множини Ω_i часткових експонент $\Omega_i = \{P_{i,0}, P_{i,1}, \ldots P_{i,Y}\}$, $\forall\, Y \in \{0,1,\ldots, 2^a\text{-}1\}$ у відповідності до всіх 2^a варіантів a-розрядних двійкових кодів Y: $Y = y_1 \cdot 2^0 + y_2 \cdot 2^1 + \ldots + y_m \cdot 2^{a-1}$, де $\forall\, m \in \{1,2,\ldots, a\}$: $y_m \in \{0,1\}$. Значення $W_{i,Y}$ обчислюються за наступною формулою:

$$\forall\, Y \in \{0,1,\ldots,\, 2^a - 1\}: P_{i,Y} = \left(W_i \cdot \prod_{m=1}^{a} A_{m+ib+(i-1)a-1}^{\,y_m}\right) \bmod M. \qquad (2.3)$$

З технологічної точки зору, найбільш ефективним є наступний порядок обчислення формули (2.3). Паралельне оброблення 2^a варіантів a-розрядних кодів експонент проводиться згідно з порядком зростання кількості одиничних бітів у їх двійкових кодах. Таким чином, кожне значення $P_{i,Y}$ обчислюється через модулярне множення одного з попередньо отриманих результатів $P_{i,0}, \ldots, P_{i,Y}$ -1 на відповідне значення A. Початковим кроком є обчислення $P_{i,0}$, що фактично дорівнює числу W_i. Далі виконуються обчислення для тих a варіантів кодів, в яких a-1 бітів дорівнюють нулю, а один біт – одиничний. Для кожного з цих кодів виконується модулярний добуток числа $P_{i,0}$ з відповідним одиничним розрядом значення A. За подібним принципом обробляються всі інші варіанти кодів експоненти до отримання множини значень $\Omega_i = \{P_{i,0}, P_{i,1}, \ldots P_{i,Y}\}$. Після обробки i-го сегмента, множина Ω_i з 2^a значень передається на

термінальний мікроконтролер. На ньому виконується модулярне множення змінної R з правильним числом $P_g \in \{P_{i,0}, P_{i,1}, \dots P_{i,Y}\}$, де g відповідає коректним значенням коду експоненти: $g = s_{(i\text{-}1)a+i\cdot b+1}\cdot 2^0 + s_{(i\text{-}1)a+i\cdot b+2}\cdot 2^1 + \dots + s_{(a+b)i}\cdot 2^{a\text{-}1}$. Тут виконується операція: $R = R \cdot P_g \bmod M$. На останньому етапі обчислень на термінальному мікроконтролері після отримання результатів з хмари для j-го сегмента та виконання останнього модулярного добутку.

2.3 Демонстрація роботи запропонованого методу на числовому прикладі

Роботу розробленого методу можна проілюструвати у вигляді наступного числового прикладу. Нехай потрібно обчислити модулярну експоненту $D^K \bmod M = 463^{22895} \bmod 30551 = 1786$. Відповідно, у рамках прикладу, числовими значеннями є $D = 463$, $K = 22895$, $M = 30551$, де D фактично виступає в якості інформаційної компоненти, K відіграє роль закритого ключа, а M – є відкритим значенням модуля. Двійковими кодами цих значень в межах числового прикладу є $D = 111001111_2$, $K = 1011001011011111_2$, $M = 1110111010101111_2$, таким чином в межах поточного прикладу розрядність експоненти рівна 15: $n = 15$. В цьому прикладі для (n-1)=14-розрядної експоненти обираються параметри поділу: кількість сегментів j встановлюється рівною 2, а довжини розрядів – $a = 3$ та $b = 4$. На першому етапі здійснюється гомоморфне шифрування числа D шляхом його піднесення до квадрата за модулем: $B = D^2 \bmod M = 463^2 \bmod 30551 = 512$. Потім формується бітовий код S експоненти як $S = K/2 = 10110010110111_2$. Початкове значення змінної R на мікроконтролері задається як $R = D = 463$. Двійковий код S поділяється на два сегменти: молодший – 0110111_2, старший – 1011001_2. Схему поділу наведено на рисунку 2.2:

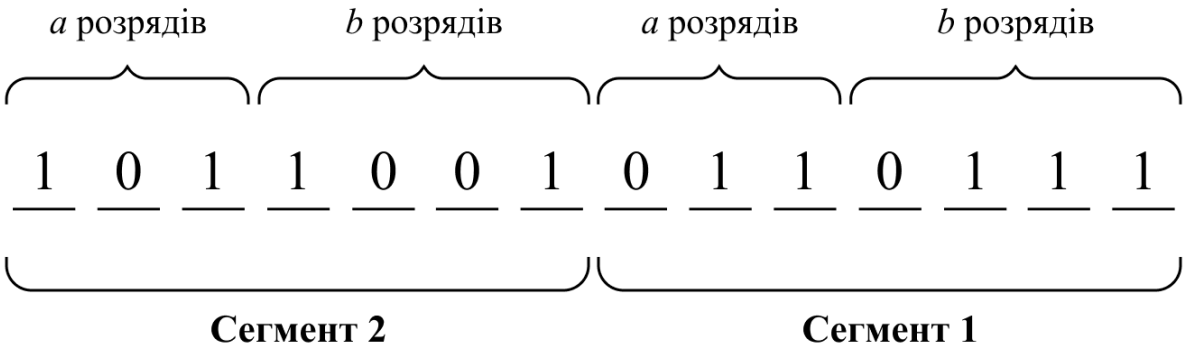

Рис. 2.2. Схема поділу коду експоненти S в межах прикладу

Із 14 бітів коду S, $j \cdot a = 6$ бітів приховані від хмарного середовища, тобто формується модифікований код $Q = 00010010000111_2$:

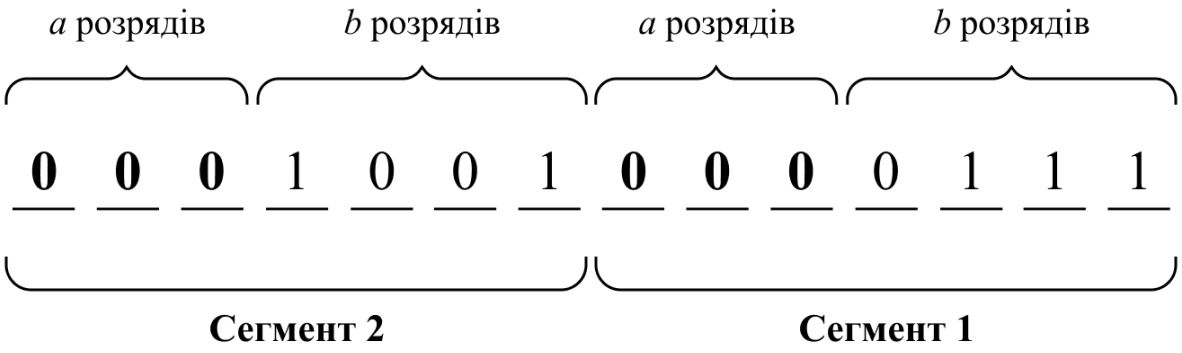

Рис. 2.3. Схема поділу модифікованого коду експоненти Q в межах прикладу

Він передається на віддалений сервер разом із числами $B = 512$ та $M = 30551$. Там значення B присвоюється змінній A_0, після чого розпочинається обробка першого сегменту коду Q. Паралельно розраховуються значення $A_1 \ldots A_7$ як модулярні квадрати. Результати обчислень наведені у вигляді таблиці 2.1:

Таблиця 2.1

Приклад обчислення $A_1,\ldots, A_7$ для першого сегменту експоненти

Номер біту експоненти Q	Результат модулярного квадрату
1	$A_1 = 512^2 \bmod 30551 = 17736$
2	$A_2 = 17736^2 \bmod 30551 = 12600$
3	$A_3 = 12600^2 \bmod 30551 = 17004$
4	$A_4 = 17004^2 \bmod 30551 = 1352$
5	$A_5 = 1352^2 \bmod 30551 = 25395$
6	$A_6 = 25395^2 \bmod 30551 = 4966$
7	$A_7 = 4966^2 \bmod 30551 = 6499$

Після цього для молодших чотирьох бітів коду Q ($q_1 = 1$, $q_2 = 1$, $q_3 = 1$, $q_4 = 0$) обчислюється число W_1 як добуток відповідних значень $A_0 \ldots A_3$, що відповідають одиничним розрядам: $W_1 = (A_0 \cdot A_1 \cdot A_2) \bmod M = (512 \cdot 17736 \cdot 12600) \bmod 30551 = 8387$. На основі W_1 обчислюється множина значень часткових експонент $\Omega_1 = \{P_{1,0}, P_{1,1}, \ldots, P_{1,7}\}$ для всіх Y із діапазону 0 до 7 (тобто 3-бітових кодів). Розрахунок ведеться у порядку зростання кількості одиниць у двійковому представленні Y: від 000_2 до 111_2. Кожне значення $P_{1,Y}$ отримується одним модулярним множенням, як зазначено в таблиці 2.2.

Таблиця 2.2

Обчислення множини Ω_1 значень $P_{1,Y}$

$Y = y_3y_2y_1$	Значення $P_{1,Y}$
000	$P_{1,0} = W_1 = 8387$
001	$P_{1,1} = (W_1 \cdot A_4) \bmod M = (8387 \cdot 1352) \bmod 30551 = 4803$
010	$P_{1,2} = (W_1 \cdot A_5) \bmod M = (8387 \cdot 25395) \bmod 30551 = 16844$
100	$P_{1,4} = (W_1 \cdot A_6) \bmod M = (8387 \cdot 4966) \bmod 30551 = 8829$
011	$P_{1,3} = (W_{1,1} \cdot A_5) \bmod M = (4803 \cdot 25395) \bmod 30551 = 12593$
101	$P_{1,5} = (W_{1,1} \cdot A_6) \bmod M = (4803 \cdot 4966) \bmod 30551 = 21918$
110	$P_{1,6} = (W_{1,2} \cdot A_6) \bmod M = (16844 \cdot 4966) \bmod 30551 = 29217$
111	$P_{1,7} = (W_{1,6} \cdot A_4) \bmod M = (29217 \cdot 1352) \bmod 30551 = 29492$

Після обчислення цієї множини, значення $P_{1,3}$, що відповідає позиціям $s_1 – s_7$ коду S, використовується для оновлення R: $R = R \cdot P_{1,3} \bmod 30551 = 463 \cdot 12593 \bmod 30551 = 25869$. Паралельно в хмарі обробляється другий сегмент Q. Обчислені значення $A_8–A_{14}$ представлені в таблиці 2.3:

Таблиця 2.3

Приклад обчислення $A_8,…, A_{14}$ для другого сегменту

Номер біту експоненти Q	Результат модулярного квадрату
8	$A_8 = 6499^2 \bmod 30551 = 15519$
9	$A_9 = 15519^2 \bmod 30551 = 5828$
10	$A_{10} = 5828^2 \bmod 30551 = 23423$
11	$A_{11} = 23423^2 \bmod 30551 = 2071$
12	$A_{12} = 2071^2 \bmod 30551 = 11901$
13	$A_{13} = 11901^2 \bmod 30551 = 29916$
14	$A_{14} = 29916^2 \bmod 30551 = 6062$

Для чотирьох старших бітів ($q_8 = 1$, $q_9 = 0$, $q_{10} = 0$, $q_{11} = 1$) визначається значення W_2 як добуток значень, які відповідають одиничним позиціям: $W_2 = (A_7 \cdot A_{10}) \bmod M = (6499 \cdot 23423) \bmod 30551 = 20995$. На базі W_2 формується множина часткових експонент $\Omega_2 = \{P_{2,0}, P_{2,1}, …, P_{2,7}\}$. Кожне з цих значень також обчислюється одним модулярним множенням, згідно з таблицею 2.4.

Таблиця 2.4

Обчислення множини Θ_2 значень P_{2Y}

$Y = y_3y_2y_1$	Значення $P_{2,Y}$
000	$P_{2,0} = W_2 = 20995$
001	$P_{2,1} = (W_2 \cdot A_{11}) \bmod M = (20995 \cdot 2071) \bmod 30551 = 6572$
010	$P_{2,2} = (W_2 \cdot A_{12}) \bmod M = (20995 \cdot 11901) \bmod 30551 = 15417$
100	$P_{2,4} = (W_2 \cdot A_{13}) \bmod M = (20995 \cdot 29916) \bmod 30551 = 18962$
011	$P_{2,3} = (W_{2,1} \cdot A_{12}) \bmod M = (6572 \cdot 11901) \bmod 30551 = 2812$

Таблиця 2.4 (продовження)

101	$P_{2,5} = (W_{2,4} \cdot A_{11}) \bmod M = (18962 \cdot 2071) \bmod 30551 = 12267$
110	$P_{2,6} = (W_{2,4} \cdot A_{12}) \bmod M = (18962 \cdot 11901) \bmod 30551 = 17076$
111	$P_{2,7} = (W_{2,6} \cdot A_{11}) \bmod M = (17076 \cdot 2071) \bmod 30551 = 16889$

Після завершення обробки другого сегмента, Ω_2 передається на мікроконтролер, де відбувається фінальна операція: $R = (R \cdot P_{2,5}) \bmod 30551 = (25869 \cdot 12267) \bmod 30551 = 1786$. Таким чином, остаточне значення $R = 1786$ є результатом модулярного експоненціювання, здійсненого за запропонованим методом розподілених обчислень між мікроконтролером і хмарним середовищем.

2.4 Оцінка ефективності запропонованого методу

Оцінка ефективності розробленого методу здійснюється на основі двох ключових показників. Першим є ступінь захисту від несанкціонованої реконструкції секретного експонентного коду у хмарному середовищі на основі отриманих даних. Другий показник ефективності – це рівень прискорення виконання операції модулярного експоненціювання завдяки використанню обчислювальних потужностей віддалених систем.

Рівень захищеності визначається необхідними обчислювальними затратами на відновлення експоненти. Однією з найпростіших і водночас потенційно результативних стратегій атаки є повний перебір усіх можливих комбінацій бітів коду. Такий підхід стає особливо актуальним у сфері Інтернету Речей, де експоненціювання часто застосовується в алгоритмах цифрового підпису, які можуть бути перехоплені при передачі мережею [41]. Тобто, зловмисник теоретично здатен отримати значення D та $L = D^K \bmod M$, прослуховуючи мережевий трафік у потенційно вразливому середовищі. На цій основі він може перевірити кожен варіант експоненти, порівнюючи результат обчислення з наявним значенням L.

Цей метод злому вимагає перебору 2^λ варіантів, де λ – це число бітів

експоненти, значення яких недоступні хмарній системі. На практиці λ задається з урахуванням особливостей застосування таким чином, щоб зробити атаку економічно невигідною через великі витрати ресурсів. У межах запропонованої схеми кількість прихованих бітів визначається як добуток параметрів a та j, тобто $\lambda = a \cdot j$. Таким чином, максимально можливе число комбінацій P становить 2^{a+b}. Необхідний рівень стійкості гарантується за умови виконання наступної умови:

$$a \cdot j = \lambda \ . \tag{2.4}$$

Цілком очевидно, що умова (2.4) відкриває можливість гнучко налаштовувати параметри реалізації – j, a та b, – що, в свою чергу, дозволяє адаптувати систему під конкретні вимоги до рівня захисту, які задаються значенням параметра λ.

Другим важливим аспектом ефективності запропонованого підходу є підвищення швидкості виконання модулярного експоненціювання. У межах цієї методики обчислення організовано як сукупність трьох взаємопов’язаних етапів: виконання обчислень на віддаленому сервері, передача отриманих результатів на термінальний мікроконтролер, а також обробка на ньому зашифрованих фрагментів експоненти. Загальна продуктивність експоненціювання у даній схемі визначається переважно швидкодією мікроконтролера, яка вважається заданою і постійною.

Таким чином, для досягнення прискорення обчислювального процесу необхідно забезпечити два ключові умови: мінімізувати обсяг обробки на мікроконтролері та організувати його безперервну роботу без простоїв. Оскільки кожен з j сегментів експоненти обробляється однаковим способом, для аналізу часових витрат усіх трьох етапів досить розглянути обробку одного з них. Якщо припустити, що час виконання операцій на віддаленому сервері більший за час пересилки даних, то узагальнену схему взаємодії процесів можна подати у вигляді, наведеному на рис. 2.4:

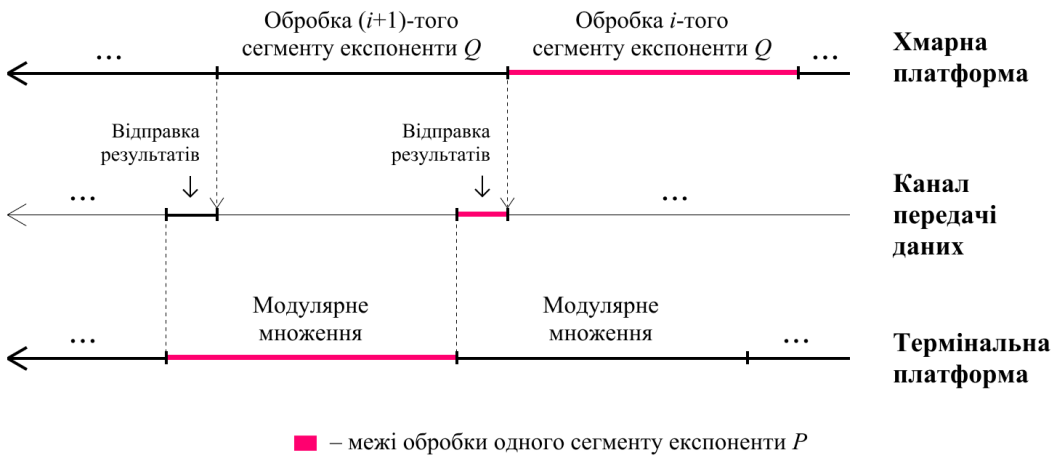

Рис. 2.4. Схема взаємодії процесів в запропонованому методі

Максимальна ефективність у часі досягається тоді, коли критичним шляхом у запропонованому методі стає виконання j модулярних множень на термінальному мікроконтролері, кожне з яких триває t_{mult}. Це відповідає кількості сегментів, на які поділено код експоненти. Щоб виконати цю умову, обчислення у хмарі повинні бути організовані таким чином, щоб завершення передачі результатів обробки чергового сегменту збігалося з моментом завершення обробки попереднього сегменту на мікроконтролері.

Для побудови адекватної моделі методу варто обґрунтувати, що в практичних сценаріях обчислення у хмарі суттєво триваліші за процес передачі результатів, а отже, останній не є обмежуючим фактором у часовій продуктивності. Відповідно до логіки методу, обробка одного сегменту експоненти Q у хмарній системі передбачає виконання $2^a + 1.5 \cdot b + a$ модулярних множень. Позначивши час виконання одного такого множення у хмарі як t_m, можна отримати загальну тривалість обробки як $(2^a + 1.5 \cdot b + a) \cdot t_m$. Більшість сучасних хмарних систем обладнано криптографічними прискорювачами, які реалізують модулярне множення апаратно. У сучасних вбудованих обчислювальних системах, зокрема в термінальних мікроконтролерах, які використовуються для виконання криптографічних операцій, дедалі ширше впроваджуються

спеціалізовані апаратні засоби – криптопроцесори. Вони являють собою окремі функціональні модулі, інтегровані у архітектуру мікроконтролера або реалізовані як автономні апаратні компоненти, що забезпечують апаратне прискорення виконання криптографічних перетворень, зокрема модулярного множення, яке є базовою операцією для алгоритмів шифрування з відкритим ключем [42]. Застосування криптопроцесорів дозволяє значно зменшити час виконання окремих операцій, які в програмній реалізації на загальнопризначених процесорах є обчислювально складними. Наприклад, згідно з технічною документацією виробника, апаратний криптографічний прискорювач CryptoSwift 600 здатен виконувати модулярне множення за 0.05 мс [24]. Такий рівень продуктивності досягається завдяки використанню паралельної обробки, конвеєрної архітектури, оптимізованих логічних схем обробки довгих цілих чисел, а також за рахунок уникнення додаткових витрат часу, характерних для програмного керування обчисленням [42, 43]. Крім високої продуктивності, криптопроцесори забезпечують і підвищену безпеку, оскільки більшість з них реалізує захист від атак сторонніми каналами, зокрема диференційного аналізу споживання енергії (DPA) або часових атак [44]. Це особливо важливо у контексті виконання частини криптографічних операцій на термінальній платформі в умовах довіреного обчислювального середовища, зокрема при реалізації розподілених криптографічних протоколів із використанням хмарних обчислень [45]. Таким чином, використання криптопроцесорів, таких як CryptoSwift 600, дозволяє забезпечити не лише високу швидкодію виконання модулярних множень, а й гарантувати безперервність обчислювального процесу на термінальному мікроконтролері. Це, своєю чергою, забезпечує виконання ключової умови досягнення часової ефективності у межах запропонованої схеми хмарного експоненціювання – час обчислень у хмарі має бути співставний або перевищувати час, необхідний для виконання одного модулярного множення на терміналі [46].

З огляду на попередній аналіз, можна вважати t_m = 0.05 мс, а типові значення параметра a знаходяться в межах від 3 до 6, а параметр b приймає відповідні значення 61, 56, 45 або 24. Для прикладу, коли a = 6 та b = 24, час обробки одного сегменту у хмарі обчислюється як: $(2 \cdot 6 + 1.5 \cdot 24 + 6) \cdot 0.05 =$ 5.3 мс. Після завершення обробки кожного сегменту в хмарі результати в обсязі 2^a значень передаються до мікроконтролера через мережу. Час передачі включає як затримку встановлення з'єднання t_{conn}, так і сам процес надсилання t_{send}. За даними [47], середній час встановлення з'єднання становить приблизно 40 мс, тоді як швидкість передавання даних через бездротовий канал – орієнтовно 100 Mbps. За умови a = 6, обсяг переданих даних складає 64 значення. Час t_{send} для такого обсягу становить приблизно 1.28 мс. З цього випливає, що навіть при використанні швидкісного зв'язку, тривалість обчислень у хмарі істотно перевищує час передавання їх результатів. Тобто затримка мережі не є критичною в оцінці ефективності. Відтак, щоб забезпечити неперервну роботу термінального мікроконтролера, необхідно, щоб час обробки одного сегменту P у хмарі $(2^a + 1.5 \cdot b + a) \cdot t_m$ не перевищував часу одного модулярного множення на мікроконтролері, тобто:

$$(2^a + 1.5 \cdot b + a) \cdot t_m \leq t_{mult} . \tag{2.5}$$

Нехай відношення часу виконання модулярного множення на термінальному мікроконтролері до часу цієї ж операції в хмарному середовищі позначається символом σ, тобто $\sigma = t_{mult} / t_m$. З урахуванням цього, можна переписати нерівність (2.5) у формі:

$$(2^a + 1.5 \cdot b + a) \leq \sigma. \tag{2.6}$$

Звідси випливає, що значення параметрів j, a та b, які визначають конфігурацію реалізації методу, повинні відповідати такій системі обмежень:

$$\begin{cases} a \cdot j \geq \psi \\ 2^a + 1.5 \cdot b + a \leq \sigma \end{cases}. \quad (2.7)$$

Значення параметрів ψ та σ залежать від конкретних умов використання алгоритму та встановлюються до початку процесу модулярного експоненціювання. Оскільки кількість невідомих у системі (2.7) перевищує кількість наявних обмежень, для її розв'язання необхідно додати ще одне рівняння. Враховуючи, що двійкове представлення експоненти *S* довжини $n - 1$ біт розбивається на j сегментів по $a + b$ біт кожен, можна доповнити систему рівнянням:

$$\begin{cases} a \cdot j \geq \psi \\ 2^a + 1.5 \cdot b + a \leq \sigma \\ j \cdot (a + b) = n - 1 \end{cases}. \quad (2.8)$$

У більшості практичних випадків величина n має велику розрядність, тому її можна апроксимувати як $n - 1 \approx n$, що дозволяє спростити третє рівняння системи до вигляду $j \cdot (a + b) = n$:

$$\begin{cases} a \cdot j \geq \psi \\ 2^a + 1.5 \cdot b + a \leq \sigma \\ j \cdot (a + b) = n \end{cases}. \quad (2.9)$$

Якщо виразити кількість сегментів j як ψ / a відповідно до першого рівняння системи (2.9), то після нескладних алгебраїчних перетворень третє рівняння може бути представлено як:

$$\psi + \frac{\psi \cdot b}{a} = n. \quad (2.10)$$

Звідси параметр b можна визначити за формулою:

$$b = \frac{n \cdot a}{\psi} - a. \quad (2.11)$$

Підставивши це значення у друге обмеження системи (2.19), його можна звести до вигляду:

$$2^a + a \cdot (1.5 \cdot \frac{n}{\psi} + 0.5) \leq \sigma \quad (2.12)$$

Оскільки відношення n / ψ у типових застосуваннях є значно більшим за одиницю, доданком 0.5 у цьому виразі можна знехтувати:

$$2^a + a \cdot \frac{1.5 \cdot n}{\psi} \leq \sigma \quad (2.13)$$

Зазначена нерівність є неявною, отже, отримати точне значення параметра a в аналітичній формі неможливо. Найбільше можливе ціле значення a, що задовольняє дану умову, знаходиться методом перебору. Нижню межу для a задають як $a \geq 1$, тоді як верхню доцільно обмежити значенням $\log_2\sigma$. Таким чином, пошук починається з найбільшого цілого числа, меншого за $\log_2\sigma$, і продовжується в напрямку зменшення до того моменту, коли буде виконано умову (2.14). Після визначення прийнятного значення a, параметри h та b обчислюються на основі першого рівняння системи (2.9) та виразу (2.11) відповідно.

Запропонований алгоритм підбору параметрів конфігурації j, a та b можна проілюструвати на конкретному числовому прикладі. Припустімо, що бітова довжина чисел, над якими виконується модулярне експоненціювання, дорівнює $n = 4096$. В умовах заданого застосування рівень криптографічної стійкості до атак, пов’язаних із потенційною спробою відновлення експоненти за перехопленими даними, визначається складністю обчислення 2^{256} операцій модулярного експоненціювання. Це означає, що витрати на здійснення такого обчислення перевищують потенційну вигоду від отримання доступу до секретного коду експоненти, що виправдовує вибір $\psi = 256$. Крім того, виходячи з продуктивності апаратних платформ, на яких реалізується обчислення, співвідношення часу виконання операції на мікроконтролері до часу обробки в хмарі приймається рівним $\sigma = 100$.

На першому етапі розв’язання задачі визначається можливе значення параметра a, що відповідає вказаним ψ та σ, а також задовольняє умову (2.13). Як зазначалося раніше, пошук ведеться в напрямку зменшення, починаючи від найбільшого цілого числа, меншого за $\log_2\sigma$. У нашому випадку це 6. Перевірка цього значення у лівій частині нерівності (2.13) дає результат: $2^6 + 6 \cdot 1.5 \cdot 4096 / 256 = 208$, що перевищує обмеження $\sigma = 100$. Наступне значення $a = 5$ призводить до результату: $2^5 + 5 \cdot 1.5 \cdot 4096 / 256 = 152$. При $a = 4$ отримуємо значення 112, а вже для $a = 3$ вираз у лівій частині набуває значення 80. Оскільки лише останній варіант задовольняє обмеження, саме $a = 3$ вважається допустимим максимальним цілим значенням, яке відповідає умові. На другому етапі виконуються обчислення параметрів b та j для знайденого значення $a = 3$. Відповідно до виразу (2.12), параметр b визначається як $b = (n \cdot a / \psi) - a = (4096 \cdot 3 / 256) - 3 = 45$. Далі, із третього рівняння системи (2.9) випливає, що $j = n / (a + b) = 4096 / (3 + 45) \approx 85$. Таким чином, параметри, які забезпечують виконання всіх необхідних умов і відповідають обраній моделі захисту, мають такі значення: $a = 3$, $b = 45$, $j = 85$.

Часове значення T_{me}, що відповідає загальній тривалості виконання модулярного експоненціювання відповідно до запропонованого підходу, визначається виключно тривалістю обчислень на мікроконтролері, тобто $T_{exp} = T_m$. У межах розробленої процедури загальний час T_m складається з трьох основних компонентів:

- часу, необхідного для проведення одного модулярного множення під час шифрування елемента D;
- часу очікування першої відповіді від віддаленої системи (який за своєю тривалістю дорівнює одному модулярному множенню);
- часу виконання j модулярних множень.

Відповідно, загальна тривалість операції модулярного експоненціювання визначається як:

$$T_{exp} = T_m = (j + 2) \cdot t_{mult}. \quad (2.14)$$

де t_{mult} – це середній час одного модулярного множення на мікроконтролері. Це значення можна вивести через співвідношення $\varphi = t_{mult} / t_m$, що визначає відносну повільність мікроконтролера в порівнянні з хмарною обробкою. Наприклад, якщо $t_m = 0{,}05$ мс і $\sigma = 100$, тоді $t_{mult} = \sigma \cdot t_m = 100 \cdot 0{,}05 = 5$ мс. Отже, для $j = 85$ (що відповідає прикладу вище), загальний час T_{exp} складає $T_{exp} = T_m = (85 + 2) \cdot 5 = 435$ мс.

Для комплексного оцінювання ефективності розробленого підходу доцільно здійснити порівняльний аналіз його часових характеристик з аналогічними відомими методами за умови однакового рівня криптографічного захисту. У цьому контексті доцільно ввести два показники:

- коефіцієнт φ_1, що відображає виграш у часі за рахунок використання зовнішніх (віддалених) обчислювальних ресурсів для прискорення процесу модулярного експоненціювання;
- коефіцієнт φ_2, що характеризує співвідношення тривалості обчислень у межах запропонованого методу до часу виконання аналогічних операцій у класичних схемах.

Показник φ_1 характеризує ефективність прискорення обчислень і визначається як відношення часу, необхідного для повного виконання модулярного експоненціювання на термінальному мікроконтролері, до тривалості обчислень за допомогою віддаленої обчислювальної інфраструктури відповідно до запропонованого підходу. Для класичного способу з обробкою експоненти, починаючи з молодших бітів, час визначається кількістю операцій модулярного множення, що становить приблизно $1{,}5 \cdot n$. Це пояснюється тим, що при кожному біті n-розрядної експоненти виконується одна операція піднесення до квадрату, а також із вірогідністю 50% – додаткове множення [48]. Таким чином, формула для розрахунку коефіцієнта φ_1 набуває вигляду:

$$\varphi_1 = \frac{(1.5 \cdot n) \cdot t_{mult}}{j \cdot t_{mult} + 2 \cdot t_{mult}} = \frac{1.5 \cdot n}{j+2} \ . \quad (2.15)$$

Для прикладу з параметрами n = 4096 та j = 85 отримується $\varphi_1 = 1.5 \cdot 4096/87 \approx 71$. Тобто реалізація обчислень за запропонованою схемою дозволяє скоротити час модулярного експоненціювання приблизно у 71 раз. За результатами практичного тестування, реальна економія часу склала коефіцієнт 67, що практично співпадає з отриманою теоретичною оцінкою.

Коефіцієнт φ_2 вводиться для оцінки переваги розробленого методу у порівнянні з іншими сучасними підходами, які також використовують розподілену обробку даних із залученням зовнішніх систем. Для формалізації φ_2 обирається один із найефективніших наявних методів, зокрема той, який ґрунтується на розкладанні експоненти у мультиплікативно-адитивну форму [32]. У згаданому методі загальна кількість операцій модулярного множення приблизно дорівнює $1.5 \cdot \varepsilon + k + 2$, де ε – кількість бітів експоненти, що обробляються на локальному мікроконтролері, а k – кількість частин, на які розбито експоненту. Тоді співвідношення φ_2 обчислюється як:

$$\varphi_2 = \frac{(1.5 \cdot \varepsilon + k + 2) \cdot t_{mult}}{j \cdot t_{mult} + 2 \cdot t_{mult}} = \frac{1.5 \cdot \varepsilon + k}{j+2} \ . \quad (2.16)$$

Припустимо, що для забезпечення криптографічної стійкості необхідно здійснити перебір 2^{256} варіантів, тоді в методі [32] ε = 128, k = 54. Підставляючи ці значення та приймаючи n = 4096 і j = 85, отримується $\varphi_2 = (1.5 \cdot 128 + 54)/85 \approx 3$. Це означає, що представлений метод дає змогу зменшити час виконання модулярного експоненціювання приблизно втричі порівняно з найкращими існуючими схемами. Такий ефект досягається завдяки раціональному перерозподілу навантаження: обсяг складних обчислень переноситься на потужні зовнішні ресурси, що дозволяє зменшити час критичного шляху на обмежених можливостях термінального пристрою.

Висновки до розділу 2

У другому розділі магістерської дисертації, спрямованого на підвищення ефективності виконання критично важливої криптографічної операції з відкритим ключем – модулярного експоненціювання – на термінальних пристроях Інтернету Речей (IoT) із використанням захищених хмарних обчислень, досягнуто наступних результатів.

1. На теоретичному рівні обґрунтовано, що одним із ключових шляхів підвищення ефективності є зменшення обчислювального навантаження на обмежені в ресурсах термінальні мікроконтролери. Такий підхід дозволяє суттєво скоротити загальний час виконання операції за рахунок делегування складних обчислень на потужні віддалені ресурси.
2. Розглянуто та проаналізовано можливі шляхи вирішення проблеми підвищення ефективності реалізації модулярного експоненціювання на термінальних пристроях Інтернету Речей.
3. Теоретично обґрунтовано, розроблено та досліджено метод реалізації модулярного експоненціювання, що базується на безпечному розподілі обчислень між термінальним пристроєм та хмарною інфраструктурою. Суть методу полягає в тому, що замість послідовного виконання модулярних множень на термінальному мікроконтролері відповідно до розрядів експоненти, система попередньо обчислює всі можливі варіанти на хмарній системі. Після цього термінальний пристрій лише обирає необхідні результати згідно зі своїм ключем. Це дозволяє значно скоротити кількість обчислень, які мають бути виконані безпосередньо на пристрої користувача, що, в свою чергу, призводить до пришвидшення всієї процедури обчислення експоненти.
4. Розроблено математичну модель процесу такого розподіленого обчислення, яка дозволяє раціонально підібрати ключові параметри методу з урахуванням характеристик як термінального пристрою, так і віддаленого середовища.

5. Результати теоретичних розрахунків та проведених експериментів узгоджуються між собою та свідчать про значне підвищення продуктивності. Зокрема, реалізація запропонованого методу забезпечує прискорення виконання операції модулярного експоненціювання в середньому в 3–4 рази порівняно з існуючими методами, що також передбачають залучення хмарних обчислень. Це підтверджує доцільність застосування розробленого підходу в практичних системах криптографічного захисту для IoT-середовищ, де ресурси термінальних пристроїв є обмеженими.

РОЗДІЛ 3. РОЗРОБКА МЕТОДУ БЕЗПЕЧНОГО ЗАЛУЧЕННЯ ХМАРНИХ СИСТЕМ ДЛЯ ПРИСКОРЕНОЇ РЕАЛІЗАЦІЇ КРИПТОГРАФІЇ З ВІДКРИТИМ КЛЮЧЕМ В ІНТЕРНЕТІ РЕЧЕЙ

3.1 Розробка методу безпечного залучення хмарних систем для прискореної реалізації криптографії з відкритим ключем в Інтернеті Речей

Запропонований підхід базується на подальшому розвитку ідеї мультиплікативно-адитивного представлення ключового коду у вигляді $K = U \cdot F + W$, яка отримала широке застосування як засіб підвищення ефективності обчислення експоненти на енергообмежених термінальних пристроях із залученням зовнішніх обчислювальних ресурсів [49]. Один з можливих шляхів підвищення рівня захищеності обчислень полягає у використанні особливих циклічних властивостей операції модулярного експоненціювання. На етапі генерації криптосистеми, термінальному мікроконтролеру, як одному з учасників процесів моніторингу та керування об’єктами реального світу на базі Інтернету Речей, присвоюється унікальний незмінний закритий ключ K. В рамках операції модулярного експоненціювання цей ключ використовується для шифрування компоненти D. Зокрема при реалізації механізмів цифрового підпису – одного з найбільш важливих елементів забезпечення цілісності та автентичності даних, які передаються системі керування – ключем K здійснюється шифрування хеш-сигнатури повідомлення D для підтвердження автентичності мікроконтролера, який надсилає це повідомлення [50].

Принципова відмінність такого підходу має на меті підвищення ефективності хмарних обчислень і полягає у переході від хмарного обчислення $P = B^F \bmod M$ до послідовного обчислення $Y = B^Q \bmod M$ в хмарі та $P = Y^g \bmod M$ на мікроконтролері так, щоб $g << F$. Такий підхід дозволяє уникнути необхідності відправки на віддалену систему прямої складової F закритого ключа K і реалізується ґрунтуючись на циклічних властивостях модулярного експоненціювання. З огляду на те, що генерація ключів здійснюється на етапі

створення системи моніторингу чи управління, вказані властивості є відомими тому, хто створював таку систему. Формально ці властивості представляються у вигляді малої теореми Ферма та її узагальнення Ейлером.

Нехай через C позначається період повторень, такий щоб:

$$\forall\, B < M : B^C \bmod M = 1\ . \tag{3.1}$$

Якщо M – просте число, то $C = M$ - 1, а у випадку, коли M утворюється як добуток двох простих чисел $M = p \cdot q$, то $C = (p - 1)(q - 1)$. З огляду на те, що модуль M є відкритим значенням, число C представляється секретним елементом лише у другому з описаних вище випадків, а саме коли потребує факторизації модуля M для отримання доступу до значень p та q [51]. Тоді, з урахуванням формули (3.1), операція обчислення модулярної експоненти R може бути представлена формулою:

$$P = B^F \bmod M = (B^F \bmod M \cdot B^{jC} \bmod M) \bmod M = B^{jC+F} \bmod M\ , \tag{3.2}$$

де j – кількість періодів, яка є довільним цілим числом. При реалізації переходу, умовою забезпечення еквівалентності результатів є дотримання формули (3.3):

$$P = (B^Q \bmod M)^g \bmod M = B^F \bmod M\ . \tag{3.3}$$

Відповідно, на основі формул (3.2) та (3.3), можна скласти базове в рамках розробленого методу рівняння :

$$Q \cdot g = j \cdot C + F\ . \tag{3.4}$$

Фактично, мається на увазі використання зовнішнього мультиплікативно-адитивного розкладення $K = U \cdot F + W$ коду ключа K лише для шифрування інформаційної складової D та забезпечення ефективного підбору параметрів гомоморфного шифрування. Іншими словами, обчислення на мікроконтролері $B = D^U \bmod M$ виключає доступ віддалених систем до реального значення коду D.

Водночас головна особливість, яка безпосередньо впливає на рівень захищеності обчислень, полягає у способі представлення коду F експоненти. Фактично мова йде про більш складний формат мультиплікативно-адитивного m-рівневого розкладення модифікованого коду експоненти F на складові Q_1, Q_2,…, Q_m, які оброблюються віддалено, та компоненти g_1, g_2,…, g_{m+1}, які оброблюються на мікроконтролері:

$$F + j \cdot C = \sum_{i=1}^{m} \left(h_i \cdot \prod_{j=i}^{m} Q_j \right) + h_{m+1} \ , \qquad (3.5)$$

де j виступає в якості випадкової цілої складової, а число C – секретним параметром криптосистеми. Число m визначає кількість ітерацій рекурсії та обирається у відповідності до конкретного практичного застосування [52]. Зокрема для значення $m = 3$, формула (3.5) представлення модифікованого коду F має наступний вигляд:

$$F + j \cdot C = (((g_1 \cdot Q_1 + g_2) \cdot Q_2 + g_3) \cdot Q_3 + g_4 \ . \qquad (3.6)$$

Використання складової $j \cdot C$ у модифікованому представленні коду F експоненти дозволяє значно збільшити діапазон значень лівої сторони тотожності (3.6) без впливу на результат обчислень. Це дає можливість гнучко і в широких межах генерувати значення складових Q_1, Q_2,…, Q_m та g_1, g_2,…, g_{m+1} коду $F + j \cdot C$. Такий формат розкладення модифікованого коду F експоненти передбачає можливість динамічного регулювання розрядностей всіх компонент g в рамках незмінності їхньої сумарної довжини H. За рахунок такої особливості, об’єм обчислень на термінальному мікроконтролері може бути зменшений при однаковому обсязі перебору для несанкціонованого відновлення компонент g секретного коду експоненти.

В рамках певного мультиплікативно-адитивного розкладення секретного коду експоненти K, реальне значення коефіцієнту β прискорення модулярного експоненціювання за рахунок залучення віддалених комп’ютерних потужностей залежить від організації розподіленого обчислювального процесу.

Теоретично, швидкодія віддалених комп'ютерних систем на порядки більша за аналогічний показник термінальних мікроконтролерів, тому коефіцієнт β визначається часом T_{TM} роботи останніх. Нижня границя T_{TM} залежить від сумарної кількості H розрядів експоненти K, обробка яких здійснюється на термінальному мікроконтролері. В свою чергу, значення H визначається виходячи з об'єму V перебору, який має здійснити зловмисник для незаконної реконструкції секретного коду K за даними, які передаються мікроконтролером на віддалену систему.

Такий підхід дозволяє зменшити обчислювальні навантаження на термінальний мікроконтролер при зафіксованому значенні V за рахунок комбінаторного збільшення числа варіантів перерозподілу кількості розрядів компонентів g_1, g_2,…, g_{m+1}. Для оцінки зменшення обчислювального навантаження на термінальний мікроконтролер, яке визначається значенням H, при фіксованому значенні V можна використати коефіцієнт γ. Чисельне значення γ обчислюється як співвідношення $H_{MA}(V)$ кількості розрядів коду експоненти що оброблюється на термінальному мікроконтролері при традиційному мультиплікативно-адитивному розкладені до аналогічного показника $H_m(V)$ для m-рівневого розкладення коду експоненти F:

$$\gamma(V) = \frac{H_{MA}(V)}{H_m(V)} \ . \qquad (3.7)$$

Для традиційного мультиплікативно-адитивного розкладення показник $H_{MA}(V)$ розраховується як $H_{MA}(V) = \log_2(V)$. В свою чергу значення $H_m(V)$ залежить від об'єму V перебору всіх варіантів перерозподілу розрядностей компонент g_1, g_2,…, g_{m+1} для конкретного рівня m рекурсії розкладення коду $F + j{\cdot}C$ експоненти. Зокрема при однорівневій рекурсії розкладення $F + j{\cdot}C = g_1{\cdot}Q + g_2$ значення V обчислюється як кількість можливих значень g_1 та g_2 для всіх варіантів розподілу зі збереженням їх сумарної довжини $\leq H$. В першому з варіантів розподілу кількості розрядів H, які оброблюються на мікроконтролері,

між компонентами g_1 та g_2 довжина $h_1 = 0$, а довжина $h_2 = H$. Оскільки при єдиному можливому в такому випадку значенні g_1=0, мультиплікативна компонента $g_1 \cdot Q$ також приймає значення 0, що протирічить концепції суміжних між хмарою та мікроконтролером обчислень [53], значення g_1 встановлюється в 1, що не впливає на результат. При цьому адитивна компонента g_2 може приймати будь-які значення у межах розрядності від 1 до h_2. Тоді об'єми перебору можливих значень у першому варіанті розподілу розрядностей складають відповідно $v_1 = 2^{l_1}$ для компоненти g_1 та $v_2 = 2^{l_2}$ для g_2. Загальний об'єм перебору можливих значень компонент g_1 та g_2 визначається як $V_1 = v_1 \cdot v_2$, тобто у першому варіанті розподілу розрядностей $V_1 = 2^{l_1} \cdot 2^{l_2} = 2^H$. У наступному варіанті розподілу загальної розрядності H між кодами g_1 та g_2 у однорівневій рекурсії мультиплікативно-адитивного розкладення $F + j \cdot C$ довжини цих кодів дорівнюють відповідно $h_1 = 2$ та $h_2 = H - 2$. Кількість можливих значень при цьому складає $v_1 = 2^{l_1 - 1}$ для компоненти g_1 та $v_2 = 2^{l_2}$ для g_2. Таким чином, загальний об'єм перебору V_2 в даному випадку складає $V_2 = 2^{l_1 - 1} \cdot 2^{l_2} = 2^{H-1}$. За аналогічним попередньому варіанту розподілу принципу, визначається уся множина значень V_3, V_4, …, V_H для інших H-2 варіантів. Тоді загальний об'єм перебору V для однорівневої рекурсії розкладення коду $F + j \cdot C$ обчислюється як сума $V = V_1 + V_2 + \ldots + V_H$ та має вигляд:

$$V = 2^{H-1} \cdot (H + 1) . \qquad (3.8)$$

Ще один підхід до захисту бітового представлення експоненти полягає в її трансформації через об'єднання частин (конкатенацію). Суть методу полягає в поділі коду експоненти на кілька частин за принципом:

$$K = K_i \text{-} 1 \cdot 2^i + K_i \text{-} 2 \cdot 2^g + \ldots + K_1 \cdot 2^e + K_0.$$

У найпростішій конфігурації маємо лише два фрагменти, тобто $i = 2$, і тоді запис експоненти набуває вигляду $K = W \,||\, L$, де W – старші біти, а L – молодші.

Для прикладу, експоненту $K = 101011111_2 = 351_{10}$ можна розділити навпіл. У такому випадку віддалений сервер обчислює частину степеня за старшими бітами: $A = D^W \bmod M$, а на локальному пристрої (мікроконтролері) відбувається обчислення залишку від зведення до степеня лише для молодших бітів L. Нехай $K = 101011111_2 = 351$, модуль $M = 43$, а основа степеня $D = 12$. Обчислення дає результат: $D^K \bmod \mathrm{M} = 12^{351} \bmod 43 = 2$. Вибирається множина позицій бітів експоненти, які будуть оброблятися локально: $\psi = \{3, 5, 6, 9\}$, значення в цих позиціях: $\Theta = \{1, 1, 0, 1\}$. Після обнуління зазначених бітів формується частина експоненти для обробки на сервері: $\mathrm{W} = 001001011_2 = 75$. У хмарному середовищі обчислюється: $R = A^W \bmod M = 12^{75} \bmod 43 = 32$. Також сервер поетапно повертає додаткові результати, зокрема: $A_1 = 12^4 \bmod 43 = 10$, $A_2 = 12^{16} \bmod 43 = 24$, $A_3 = 12^{32} \bmod 43 = 17$, $A_4 = 12^{256} \bmod 43 = 10$. Мікроконтролер обробляє дані наступним чином: початково $Q = A_1 = 10$, далі: $Q = A_2 \cdot Q \bmod 43 = 24 \cdot 10 \bmod 43 = 25$, $Q = A_4 \cdot Q \bmod 43 = 10 \cdot 25 \bmod 43 = 35$. Остаточний результат визначається як добуток з результатом з хмари: $Q = Q \cdot R \bmod M = 35 \cdot 32 \bmod 43 = 2$, що збігається з правильним значенням.

Однак у віддаленому зведенні до степеня можна маскувати лише біти, які дорівнюють одиниці. Це створює потенційну загрозу, адже зловмисник може, наприклад, підставити одиниці замість відсутніх значень.

Для ілюстрації роботи підходу: $M = 43$, $D = 12$, експонента $K = 1011_2 = 11$, $D^K \bmod M = 12^{11} \bmod 43 = 26$. У хмару відправляється змінений код $W = 1111_2$, відповідно: $12^{15} \bmod 43 = 2$. Для отримання точного результату потрібно було б поділити його на 12^4, тобто: $12^{15} / 12^4 = 12^{11}$, але пряма операція ділення в модульній арифметиці недоступна. Тому використовується обернене значення: $D^{-4} \bmod 43 = D^{(43-1-4)} \bmod 43 = D^{38} \bmod 43$, що у нашому випадку дає: $12^{38} \bmod 43 = 13$.

Підвищити ефективність можливо шляхом симетричної перестановки бітів у представленні експоненти. Візьмемо, наприклад, модуль M, який дорівнює 19. Припустимо, обрано другий біт у представленні експоненти ($\varepsilon =$

2). Його циклічно інверсна позиція визначається як μ = 18 - 2 = 16. Якщо експонента має значення K = 00011, тобто 3 у десятковому представленні, тоді створюється новий код W = 10101, що дорівнює 21. Обчислення 12^3 mod 19 дає 18, як і обчислення 12^{21} mod 19 – тобто результат збігається, і застосовувати гомоморфне шифрування немає потреби. Сенс гомоморфного шифрування в тому, що одиницю переносять на позицію, яка є інверсною в межах модуля. У наведеному прикладі одиниця з місця ε = 2 переміщена на позицію μ = 16. Для модуля 19 можливо визначити кілька пар позицій, що є інверсними: <1,17>, <2,16>, <4,14>, <8,10>. Наприклад, пара <8,10> вказує на те, що D у степені 8 mod 19 та D у степені 10 mod 19 є взаємно інверсними. Знову, нехай K = 00011 = 3. Якщо реалізувати гомоморфне кодування за допомогою пари <8,10> і додати до K число 18, результат залишиться тим самим: $12^3 + 18 \bmod 19 = 12^{21} \bmod 19 = 18$.

Розглянемо тепер постійну експоненту K = 1010111101, що відповідає числу 701. Щоб її зашифрувати, значення D зменшується на одиницю і ділиться навпіл, тобто $K = U \cdot 2 + 1$. У цьому прикладі U = 101011110. Потім код U ділиться на три частини: W = 100010110 та Q = 001001000. В цьому розподілі експонента, що відповідає W, обчислюється у хмарному середовищі, а та, що пов’язана з Q – локально на мікроконтролері. Через кожні три ітерації хмара повертає нове значення D, а наприкінці – значення R. Для обчислення у хмарі використовується база $B = D^2 \bmod M$. Наприклад, при M = 743 та D = 529 отримаємо $B = 529 \cdot 529 \bmod 743 = 473$.

Під час першої ітерації у хмарі обчислюється наступне: $R_1 = 1$, $A_2 = B \cdot B \bmod 743 = 473 \cdot 473 \bmod 743 = 86$; далі $R_2 = 86$, $A_3 = 86 \cdot 86 \bmod 743 = 709$; після цього $R_3 = R_2 \cdot A_3 \bmod 743 = 86 \cdot 709 \bmod 743 = 48$, а $A_4 = 709 \cdot 709 \bmod 743 = 413$. Це значення A_4 (413) повертається мікроконтролеру. У першому циклі сам мікроконтролер обчислень не виконує.

Протягом другої ітерації хмара виконує такі обчислення: $R_4 = 48$, $A_5 = 413 \cdot 413 \bmod 743 = 422$; далі $R_5 = R_4 \cdot A_5 \bmod 743 = 48 \cdot 422 \bmod 743 = 195$; $A_6 = 422$

· 422 mod 743 = 507; $R_6 = R_5 = 195$, $A_7 = 507 \cdot 507 \bmod 743 = 714$. Потім $A_7 = 714$ передається мікроконтролеру, який обчислює: $Y = 1 \cdot A_4 = 413$.

У третьому циклі хмарне середовище виконує: $R_7 = 195$, $A_8 = 714 \cdot 714 \bmod 743 = 98$; $R_8 = 195$, $A_9 = 98 \cdot 98 \bmod 743 = 688$; $R_9 = R_8 \cdot A_9 \bmod 743 = 195 \cdot 688 \bmod 743 = 420$. Потім це значення ($R_9 = 420$) передається назад на мікроконтролер. Далі мікроконтролер виконує обчислення: $Y = Y \cdot A_7 \bmod 743 = 413 \cdot 714 \bmod 743 = 654$.

На завершальному етапі пристрій обчислює остаточний результат: $Y \cdot R_9 \cdot D \bmod M = 654 \cdot 420 \cdot 529 \bmod 743 = 182$. Перевірити результат можна також обчисленням $D^K \bmod M = 529^{701} \bmod 743 = 182$ – що збігається з правильним результатом.

Несанкціонована особа повинна вгадати значення перших γ бітів кожного з фрагментів, що еквівалентно перебору серед 2^γ можливих комбінацій. У той же час, обчислювальний блок на стороні мікроконтролера виконує в середньому $\gamma/2$ операцій множення [54]. Показник співвідношення ρ, який характеризує ефективність – це частка кількості всіх можливих варіантів до кількості множень, що виконуються пристроєм.

$$\rho_1 = \frac{2^{\gamma+1}}{\gamma}. \tag{3.9}$$

У випадку аналізу двох молодших бітів у кожному фрагменті, кількість можливих конфігурацій дорівнює 4^γ або $2^{2\gamma}$. Середня кількість модулярних множень на мікроконтролері на фрагмент обчислюється як середньозважене: $0.25 \cdot 1 + 0.25 \cdot 2 + 0.25 \cdot 3 = 2.25$. Таким чином, загальна середня кількість множень дорівнює $2.25 \cdot \gamma$. Відповідно, коефіцієнт ρ у цьому випадку визначається як частка між 4^γ і $2.25 \cdot \gamma$:

$$\rho_2 = \frac{2^{2\cdot\gamma}}{2.25\cdot\gamma} \approx \frac{2^{2\cdot\gamma-1}}{\gamma}. \tag{3.10}$$

Якщо порівнювати ефективність між обробкою одного біта та двох бітів, вводиться коефіцієнт η – як відношення двох варіантів ефективності.

$$\eta = \frac{\rho_2}{\rho_1} = \frac{2^{2\cdot\gamma-1}}{2^{\gamma+1}} = 2^{\gamma-2} \tag{3.11}$$

Таким чином можна помітити, що використання двох бітів дає кращі результати при кількості фрагментів, що перевищує два.

Для ілюстрації: при $\gamma = 16$, кількість комбінацій дорівнює $4^{16} \approx 4.295 \cdot 10^9$. Мікроконтролер у цьому випадку виконує близько 36 операцій множення ($2.25 \cdot 16$). Тоді $\rho_2 \approx 1.19 \cdot 10^7$. Якщо брати лише один молодший біт у фрагменті, кількість комбінацій зменшується до 2^{16}, а обчислювальна вартість – до 8 множень. Відповідно, $\rho_1 = 8192$, а коефіцієнт $\eta = \rho_2 / \rho_2 \approx 14564$. Для порівняння, теоретичне значення η, розраховане за формулою, становить $2^{14} = 16384$.

Додатково, ефективність можна збільшити шляхом асинхронного надсилання з мікроконтролера запитів на повернення поточних значень A з хмарної частини, що зменшує середню кількість множень на пристрої.

На відміну від варіантів з лінійним (адитивним) розкладенням коду експоненти, у запропонованій схемі джерелами непередбачуваності є два фактори:

- які саме операції виконує термінальний пристрій під час обчислення модулярного експоненціювання;
- в яких ділянках бітового поля експоненти ці дії розташовані.

У загальному випадку ступінь захисту такої схеми залежить від кількості бітів, що обробляються на пристрої, а також від їхньої позиції у структурі експоненти. Якщо виконання не призупиняється, ці біти зазвичай знаходяться на початку або в кінцевій частині коду експоненти.

У цьому підході застосовується фіксований код G довжиною k бітів. Розглянемо приклад: необхідно обчислити $2024^{3573} \bmod 3953 = 509$. У цьому випадку $D = 2024$, модуль $M = 3953$, а експонента K дорівнює 110111110101 у двійковій формі, тобто 3573 у десятковій. Наперед обчислюють допоміжні

значення: $u_6 = D^6 \bmod M = 2024^6 \bmod 3953 = 612$; $u_5 = D^5 \bmod M = 2024^5 \bmod 3953 = 3633$; $u_7 = D^7 \bmod M = 2024^7 \bmod 3953 = 1399$. Основне обчислення відбувається поетапно: $u_6^{512} \bmod 3953 = 2476$; $u_7^{64} \bmod 3953 = 2970$; $2476 \cdot u_6^8 \bmod 3953 = 1030$; а кінцевий результат дорівнює: $2476 \cdot 2970 \cdot 1030 \cdot 3633 \bmod 3953 = 509$.

Використовується єдиний код G, наприклад 3782. Загальний степінь піднесення дорівнює сумі показників: 512 + 64 + 8 + 1 = 585. У процесі експоненціювання від старших до молодших розрядів спочатку обчислюється $R = 612^8 \bmod 3953 = 1030$, далі $R = 1030^8 \bmod 3953 = 2426$, потім $R = 2426^2 \bmod M = 643$, і нарешті $R = 643 \cdot 2024 \bmod 3953 = 895$.

Основна ідея полягає в тому, щоб розбити код K довжиною n бітів на i рівних частин, кожна з яких має n/i бітів. Таким чином, код представлено як суму фрагментів: $K = K_1 + K_2 \cdot 2^{n/\text{i}} + K_3 \cdot 2^{2n/\text{i}} + \ldots + K_\text{i} \cdot 2^{(i\text{-}1)n/i}$. На зовнішніх обчислювальних ресурсах проводиться паралельне модулярне експоненціювання для m різних значень: $F_1, F_2, \ldots, F_m$. Кожен з цих показників також розбивається на i частин, для всіх $k \in \{1, 2, \ldots, m\}$: $F_k = F_{k,1} + F_{k,2} \cdot 2^{n/i} + \ldots + F_{k,i} \cdot 2^{(i\text{-}1)n/i}$. Визначається i множин: $\Theta_1, \Theta_2, \ldots, \Theta_i$, кожна з яких включає певні номери від 1 до m таким чином, щоб виконувалася задана умова (3.12).

$$\forall j \in \{1,2,\ldots,m\}: K_j = \sum_{q \in \Theta_j} F_{q,j} \; . \qquad (3.12)$$

Теоретичне значення коефіцієнта прискорення у цьому підході не перевищує 1.5. Це обмеження пов’язане з тим, що операції, які розпаралелюються, не належать до критичного шляху алгоритму експоненціювання за допомогою молодших бітів коду експоненти [55].

В альтернативній реалізації, тобто при експоненціюванні за алгоритмом зі старших бітів експоненти K, віддалено виконуються обчислення значень:

- $h_4 = u_4^{512} \bmod 523 = 4735^{12} \bmod 523 = 160$
- $h_2 = u_2^8 \bmod 523 = 135^8 \bmod 523 = 209$

- $h_3 = u_3^{64} \bmod 523 = 462^{64} \bmod 523 = 462$.

Підсумкове значення визначається так: $R = u_1 \cdot h_2 \cdot h_3 \cdot h_4 \bmod M = 173 \cdot 209 \cdot 462 \cdot 160 \bmod 523 = 361$. Якщо ж модулярне експоненціювання $D^K \bmod M$ виконується безпосередньо на мікроконтролері, то на ньому обчислюються значення виду: $u_1 = 1$, $u_2 = D^2 \bmod M$, $u_3 = D^3 \bmod M$, …, $u_\varphi = D^\varphi \bmod M$.

Зокрема, для прикладу $3482^{3749} \bmod 3953$, процес виглядає наступним чином:

- $k = 1$: $R = R \cdot h_1 \bmod M = 1 \cdot 3423 = 3423$
    - $R = R^8 \bmod M = 3423^8 \bmod 3953 = 60$
- $k = 2$: $R = R \cdot h_2 \bmod M = 60 \cdot 2302 \bmod 3953 = 3718$
    - $R = R^8 \bmod M = 3718^8 \bmod 3953 = 709$
- $k = 3$: $R = R \cdot h_3 \bmod M = 709 \cdot 237 \bmod 3953 = 2007$
    - $R = R^8 \bmod M = 2007^8 \bmod 3953 = 3010$

Отже, за результатами аналізу застосування розподілу при обчисленні модулярної експоненти можна зробити висновок: представлений підхід придатний як для захищеної інтеграції віддалених обчислювальних потужностей у процес експоненціювання на обмежених апаратних платформах, так і для організації паралельної обробки в середовищах із багатоядерною архітектурою. Більше того, навіть в умовах одноядерної обчислювальної системи метод демонструє перспективну ефективність.

Під час реалізації розподіленого алгоритму доцільно віддати перевагу варіанту, що аналізує біти коду експонети K у порядку від старших до молодших. Така конфігурація дозволяє уникнути дублювання обчислень і забезпечує більш ефективне використання ресурсів як мікроконтролера, так і хмарної системи.

Доцільно також розглянути можливу концепцію захищеного модулярного експоненціювання $D^K \bmod M$ на віддалених обчислювальних потужностях, яка базується на мультиплікативно-адитивному розкладені секретного коду експоненти K, тобто представлені її у вигляді $K=F \cdot a+k$. При цьому на віддалену систему передається код F, а компоненти k і a

використовуються тільки на термінальному мікроконтролері. Основною перевагою запропонованого підходу є можливість одночасного вирішення двох задач: практичне унеможливлення незаконної реконструкції секретних кодів експоненти K та числа D по коду F та відкритого значення модулю M.

Розкладення експоненти здійснюється шляхом підбору двох випадкових параметрів a і k. Причому число a обирається таким чином, щоб його розрядність m_a не перевищувала 5, а величина k має задовольняти умові $(K-k) \bmod a = 0$. Оскільки код експоненти K є частиною закритого ключа, то в реальних протоколах захисту інформації з відкритим ключем числа K і M змінюються рідко, тому їх можна вважати постійними. Це дозволяє виконувати описаний вище підбір параметрів a і k лише один раз при генерації ключів.

Такий метод захищеного віддаленого обчислення модулярної експоненти $D^K \bmod M$ передбачає наступну послідовність дій:

1. На термінальному мікроконтролері виконується операція модулярного експоненціювання $G=D^a \bmod M$.
2. Результат обчислення G, а також числа F і M відправляються на віддалену комп'ютерну систему.
3. В хмарі реалізується обчислення модулярної експоненти $W= G^F \bmod M$ і результат повертається мережою Інтернет на термінальний мікроконтролер.
4. Одночасно на термінальному мікроконтролері здійснюється обрахунок модулярної експоненти $Y=D^k \bmod M$.
5. Після отримання з хмари значення W, на термінальному мікроконтролері обчислюється модулярний добуток величин W і Y : $W \cdot Y \bmod M$.

Робота методу захищеного модулярного експоненціювання у хмарі на основі мультиплікативно-адитивного розкладення експоненти може бути проілюстрована наступним числовим прикладом. Нехай на етапі генерації криптосистеми з відкритим ключем створено сформовано модуль M=143, а також відкритий ключ Q=7 та закритий ключ K=103. Останній

використовується в якості секретного ключа термінального мікроконтролера системи віддаленого управління обєктами реального світу. Наступним кроком є розкладення експоненти: $K=F\cdot a+k$. Нехай, обрана величина a дорівнюватиме 3. Згідно з викладеним вище, значення k має бути підібрана таким чином, щоб $(K-k)$mod $a=0$. Одним із варіантів, що задовольняє умові, може бути $k=4$, оскільки (103-4) mod 3 =0.

В рамках поточного прикладу проводиться обчислення модулярної експоненти 80^{103}mod143=115, відповідно $D=80$, $K=103$, $M=143$. Згідно з п.1 проводиться розрахунок $G=D^a \bmod M = 80^3 \bmod 143=60$. У відповідності до п.2 отримане число $G=60$ та величини $F=33$ і $M=143$ відправляються на віддалену комп'ютерну систему для обчислення модулярної експоненти $W=G^F \bmod M=60^{33} \bmod 143=125$. Результат $W=125$ повертається на термінальний мікроконтролер. Одночасно за використання термінального мікроконтролера реалізується операція модулярного піднесення числа D до степеня k: $Y=D^k \bmod M=80^4 \bmod 143=81$. Відповідно до п.5 обчислюється модулярний добуток значень W та Y: $R=W\cdot Y \bmod M=125\cdot 81$mod143=115. Кінцевим результатом обчислень є код $R=115$.

Оцінку ефективності методу доцільно проводити за двома критеріями: визначення показників швидкодії та рівня захищеності від спроб лиходія реконструювати значення експоненти K та числа D за даними, що передаються в хмару.

Рівень захисту визначається об'ємом часових ресурсів, які має затратити лиходій на спроби відновлення секретного коду експоненти та числа D. Оскільки відкритий ключ Q відомий, лиходій може визначити величини X і U такі, що $X^K \bmod M=U$ обчисливши $U^Q \bmod M = X$. Відповідно, для знаходження секретного коду експоненти найбільш доцільною тактикою лиходія є підбір невідомих йому параметрів a і k. Цілком очевидно, що час T_{CR} для реалізації такого підбору залежить від кількість можливих варіантів значень параметрів a

і k, та часу T_{EXPcl} – обрахунку модулярної експоненти в хмарі. Чисельне значення T_{CR} визначається формулою:

$$T_{CR} = 2^{m_a+m_k} \cdot T_{EXP\_CL} \ . \tag{3.13}$$

З формули (3.13) випливає, що потрібний рівень захисту, тобто об’єм часових ресурсів для його порушення, можна забезпечити шляхом відповідного вибору значення параметру k.

Технологія такого гнучкого управління рівнем захищеності при використанні запропонованого методу може бути ілюстрована наступним прикладом. Нехай, отримання доступу до закритого ключа E термінального мікроконтролера системи управління віддаленим об’єктом надає лиходієві отримати вигоди V_{SRC}, що оцінюються в 1,000,000$. Відповідно, для забезпечення надійного захисту, потрібно, щоб вартість C_{CS} об'єму витрачених на підбор параметру k ресурсів перевищувала вигоди, тобто була не меншою 1,000,000$. Припустивши, що вартість оренди часу C_{CS} віддаленої комп’ютерної системи становить 100$/година, вказана умова виконується для при значеннях T_{CR} , що визначаються з рівняння:

$$T_{CR} = \frac{V_{SRC}}{C_{CS}} = \frac{10^6}{100} = 10^4 \ \textit{годин} \ . \tag{3.14}$$

Визначення чисельного значення часу T_{EXP_CL} обчислення модулярної експоненти може бути здійснене виходячи з того, що нині практично всі комп’ютерні системи, включаючи ноутбуки мають апаратні засоби для швидкого виконання цієї важливої операції у вигляді вбудованого криптопроцесора. Зокрема криптопроцесор 7955 фірми Hi/fn забезпечує модулярне експоненціювання 2048-розрядних чисел за час T_{EXP_CL}=0.083 с.= $2.3 \cdot 10^{-5}$ годин. З урахуванням визначених числових значень T_{CR} та T_{EXP_CL}, що входять до формули (3.13), можна отримати:

$$m_k + m_a = \log_2 \frac{T_{CR}}{T_{EXP\_CL}} = \log_2 \frac{10^4}{2.3 \cdot 10^{-5}} = 29$$

Звідси кількість m_k двійкових розрядів числа k визначається як 29-5 =24. Тобто, для забезпечення визначеного в поточному прикладі рівня захисту, розрядність числа k має бути не меншою 24.

В якості показника швидкодії найбільш часто використовується коефіцієнт прискорення β, який визначається співвідношенням часу T_M обчислення модулярної експоненти на термінальному мікроконтролері до часу T_C виконання цієї операції з залученням віддалених обчислювальних потужності згідно із запропонованим методом:

$$\beta = \frac{T_M}{T_C}. \tag{3.15}$$

При обрахунку модулярної експоненти класичним способом час обчислень дорівнює T_M=1,5·n·T', де n – кількість розрядів експоненти, а T' – час модулярного множення n-розрядних чисел. За умови використання схеми Монтґомері для модулярного множення, час T' становить $2 \cdot n^2 \cdot t/r$, де r – розрядність процесора, а t – час виконання процесором однієї команди типу додавання. Таким чином формула часу обрахунку модулярної експоненти має вигляд:

$$T_M = \frac{3 \cdot n^3 \cdot t}{r}. \tag{3.16}$$

З цієї формули (3.16) зрозуміло, що існує кубічна залежність часу T_M від розрядності n.

Час T_C обчислення модулярної експоненти згідно із запропонованим методом залежить від трьох складових: тривалості виконання операції $D^a \bmod M$ на термінальному мікроконтролері, максимального значення серед часу обрахунку модулярної експоненти в хмарі та часу реалізації модулярного

піднесення числа D до степеня k, а також часу T' модулярного множення $R=W\cdot Y \bmod M$. Відповідно, величина T_C може бути представлена у вигляді формули:

$$T_C = T_{EXPa} + \max\left(T_{CL}, T_{EXP_k}\right) + T'. \quad (3.17)$$

Тривалість T_{EXPa} згідно з класичним способом обчислення модулярної експоненти становить $T_{EXPa}=1{,}5\cdot m_a\cdot T'$ [56]. Аналогічним чином час виконання операції $D^k \bmod M$ дорівнює $T_{EXPk}=1{,}5\cdot m_k\cdot T'$.

При модулярному експоненціюванні на віддаленій комп'ютерній системі час T_{CL} обчислень становить:

$$T_{CL} = 2\cdot T_{DT} + T_{EXP\_CL}\ , \quad (3.18)$$

де $2\cdot T_{DT}$ – час перенесення даних в хмару і повернення результатів, T_{EXP_CL} – час виконання $G^F \bmod M$. Фактично тривалістю обчислень в хмарі T_{EXP_CL} можна знехтувати в силу швидкого виконання розрахунків, отже $T_{CL}=2\cdot T_{DT}$.

Нехай $\max(T_{CL}, T_{EXPk})=T_{EXPk}$, тоді формулу обчислення T_C можна записати в такому вигляді:

$$T_C = 1.5\cdot (m_k + m_a)\cdot T'$$

$$T_C = 1.5\cdot (m_k + m_a)\cdot T'. \quad (3.19)$$

Тоді коефіцієнт прискорення β може бути представлений у вигляді:

$$\beta = \frac{1.5\cdot n\cdot T'}{1.5(m_k+m_a)\cdot T'} = \frac{n}{m_k+m_a}\ . \quad (3.20)$$

В рамках розлянутого вище прикладу секретний код K експоненти має розрядність рівну 2048: n=2048. Згідно з описаним вище, розрядності m_k та m_a дорівнюють 24 і 5 відповідно. Тому β = 2048 / 29 = 70.6. Тобто, в рамках розглянутого прикладу застосування розробленого методу забезпечує прискорення модулярного експоненціювання в 70.6 раз.

Іншим, більш перспективним напрямком досліджень, який покладено в

основу запропонованого методу, вбачається наступний крок у розвитку ідеї мультиплікативно-адитивного розкладення коду K експоненти. У випадку, коли мультиплікативна компонента U не дорівнює нулю, використання мультиплікативно-адитивного розкладення коду експоненти відкриває можливість реалізації гомоморфного шифрування інформаційного операнду D у виразі $D^K \bmod M$, що особливо актуально в умовах обмежених обчислювальних ресурсів IoT-пристроїв. Постійне вдосконалення Інтернет-технологій та інтегрованих схем визначає швидкий розвиток концепції Інтернету Речей та активного розширення його практичних областей застосування. Серед найбільш значущих областей, з точки зору безпеки, ми можемо висвітлити системи забезпечення безпеки та інтелектуальне відеоспостереження, віддалений моніторинг здоров'я, контроль процесів, контроль якості навколишнього середовища, а також контроль над безпілотними транспортними засобами за допомогою технології Starlink.

Основна ідея концепції Інтернету Речей, полягає у використанні мережі як середовища обміну даними в системах дистанційного керування для об'єктів реального світу, що створює ряд переваг. У той же час, існують ризики, пов'язані з використанням Інтернету у системах віддаленого моніторингу та управління. Основним є загрози інформаційній безпеці. Такі загрози поділяються на три основні категорії. Серед цих категорій можливість підробки даних про моніторинг, а також команд, що надсилаються з контролюючих пристроїв, є найбільш критичними і вимагають ретельного захисту в процесі підтвердження цілісності та автентичності даних. Криптографія з відкритим ключем, зокрема, механізми цифрових підписів, традиційно використовується для реалізації такого захисту. Основна обчислювальна операція в такій криптографії – це модулярне експоненціювання, що проводиться над числами великої бітової довжини. NIST рекомендує двійкову розрядність цих чисел як 4096. Очевидно, що це значно ускладнює реалізацію модулярного експоненціювання в режимі реального часу на мікроконтролері з низькою

потужністю, як цього вимагає більшість областей систем дистанційного керування.

Теоретично існують три можливі способи підвищення продуктивності модулярного експоненціювання на термінальних платформах. Перший з них – використовувати криптопроцесори. Другий спосіб прискорити модулярне експоненціювання – це перейти на альтернативні алгебраїчні основи, наприклад, на алгебру полів Галуа. Третій спосіб підвищення продуктивності обчислення модулярної експоненти - це залучити хмарні обчислення, що можливо через наявність радіомодему у мікроконтролері. У цьому випадку необхідно зробити практично недоцільним реконструкцію секретних елементів криптографії з відкритим ключем за даними, що передаються в хмарні системи. Усі існуючі рішення, засновані на залученні хмарних систем до обчислення модулярної експоненти на мікроконтролері, можна розділити на дві групи. Перший з них складається з методів, які використовують можливості віддалених криптопроцесорів. Такі рішення характеризуються високою швидкістю з неможливістю отримання проміжних результатів, що унеможливлює ефективну організацію розподілених обчислень між термінальними та віддаленими платформами. Друга група, відповідно, включає методи, які не передбачають використання криптопроцесорів, засновані на використанні високопотужних процесорів віддалених комп'ютерних систем. Це дозволяє організувати ефективний розподіл обчислень між хмарою та термінальним мікроконтролером через широкий обмін даними між ними. У системному плані ефективність таких методів оцінюється двома критеріями: прискорення реалізації модулярного експоненціювання на термінальному мікроконтролері через використання віддалених обчислень та рівень захисту від зловмисних спроб реконструювати секретні компоненти операції на основі даних, що передаються в хмарні системи. Перший з них, як правило, визначається співвідношенням загальної кількості N бітів експоненти до кількості бітів N_{MC}, які обробляються на мікроконтролері в методах, що залучають хмарні системи.

Другий критерій визначається обсягом перебору, який необхідно здійснити для відновлення секретних бітів N_{MC} коду експоненти. Цілком зрозуміло, що обидва індикатори залежать від кількості цифрових показників N_{MC}, які обробляються на термінальному мікроконтролері. Таким чином, підвищення ефективності модулярного експоненціювання в відомих рішеннях насправді є пошуком компромісу між прискоренням цієї операції та рівнем безпеки розрахунків. Цей компроміс може бути вирішений двома способами: за рахунок скорочення часу обробки на мікроконтролері окремого блока коду експоненти та більш складної логічної організації розподілу обчислень між хмарними та термінальними платформами, що забезпечить збільшення обсягу перебору з незмінною кількістю бітів N_{MC}. Запропонований метод – це розробка концепції мультиплікативно-адитивного розкладення секретного коду експоненти $K = U \cdot G + W$, яка широко використовується для прискорення обчислення експоненти з участю хмарних систем. Кількість L секретних бітів експоненти для хмари визначається сумою бітових довжин компонентів U і W, оброблених на мікроконтролері. Але в той же час хмара не знає конкретних значень довжин цих компонентів. Таким чином, використання мультиплікативно-адитивного розкладання коду K дозволяє через збільшення обсягу пошуку незаконного відновлення коду K у хмарі, щоб підвищити рівень безпеки на $(L+1)/2$ рази порівняно з 2^L. Виходячи з цього, виникла ідея для подальшого підвищення рівня захисту без збільшення числа L, для використання більш складних форм мультиплікативно-адитивного розкладання коду експоненти. В якості першого кроку досліджено двокомпонентне мультиплікативно-адитивне розкладення коду експоненти. Загальна довжина компонент коду експоненти, оброблених на мікроконтролері дорівнює L. Обсяг пошуку N_2, а отже, і рівень захисту, наприклад для $L = 120$ складає як $N_2 = 14520 \cdot 2117 = 1815 \cdot 2120$, тоді як для традиційного мультиплікативно-адитивного розкладення експоненти об’єм перебору становить $N_1 = 60{,}5 \cdot 2120$, а для звичайного адитивного розкладення – 2120.

Як наступний крок, досліджено трикомпонентне мультиплікативно-адитивне розкладення коду експоненти. За аналогією з двокомпонентним розкладанням, загальна довжина компонентів U_1, U_2 та W, які обробляються на мікроконтролері, дорівнює L. З трикомпонентним мультиплікативно-адитивним розкладанням для $L = 120$ об'єм перебору становить $N_3 = 590480 \cdot 2116 = 36905 \cdot 2120$, що в 20 і 610 разів більше, ніж той самий показник для двокомпонентного та традиційного однокомпонентного мультиплікативно-адитивного розкладення відповідно. Отже, порівняно зі звичайним адитивним розкладанням експоненти, використання трикомпонентного мультиплікативно-адитивного розкладання дозволяє збільшити рівень захищеності у 36905 разів.

Суть методу полягає в тому, що на початковому етапі термінальний мікроконтролер обчислює проміжний результат $B = D^U \bmod M$, який надалі передається до віддаленого хмарної системи для подальшого обчислення $B^F \bmod M$, де $K = U \cdot F + W$.

Таким чином, обчислення експоненти розділяється між локальним пристроєм і хмарною платформою, що дозволяє зменшити навантаження на обчислювальні ресурси терміналу без зниження рівня безпеки. Зокрема, мікроконтролер виконує лише ті обчислення, які не містять критичної або секретної інформації, наприклад, пов’язаної з повним значенням експоненти.

Використання розкладення $K = U \cdot F + W$ передбачає локальне обчислення двох модулярних експонент: $D^U \bmod M$ та $D^W \bmod M$. При цьому, загальна кількість операцій модулярного множення, що виконуються на пристрої, визначається сумарною довжиною H біт для обох компонент U і W. Потенційний зловмисник, який намагається незаконно реконструювати повне значення K, змушений перебирати всі можливі варіанти пар (U, W) за умови фіксованої довжини H, що значно ускладнює задачу.

Розглянувши різні сценарії довжини U, можна продемонструвати, що кількість допустимих комбінацій розкладання експоненти зростає експоненційно, а саме: у найпростішому випадку, коли $U = 1$, кількість

можливих варіантів для W дорівнює 2^H. У випадку, коли $h = 2$, тобто U займає два біти, ми маємо вже $2 \cdot 2^{H-2} = 2^{H-1}$ комбінацій. Загальна тенденція така, що при довжині $U = h$, кількість комбінацій дорівнює 2^{H-1} незалежно від значення h. Повна кількість можливих варіантів пар (U, W) з урахуванням усіх значень h дорівнює $F_1 = (H+1) \cdot 2^{H-1}$.

Це означає, що складність повного перебору можливих варіантів експоненти K зростає у порівнянні зі стандартною схемою з 2^H до $(H+1) \cdot 2^{H-1}$, що дає приблизне підвищення рівня захисту в $(H+1)/2$ разів.

Принципово важливо, що таке зростання стійкості досягається без необхідності збільшення довжини обчислюваних локально компонент. Ефективність методу забезпечується як через збільшення кількості незалежних компонент у розкладі, так і шляхом випадкової модифікації самої експоненти з урахуванням періодичних властивостей операції модулярного експоненціювання.

У найпростішій формі реалізація запропонованого методу передбачає таке розкладання: $K + j \cdot C = U_1 \cdot F_1 + U_2 \cdot F_2 + W$, де F_1 і F_2 – це частини, які обробляються на віддалених обчислювальних платформах, U_1, U_2 та W – компоненти, які обчислюються безпосередньо на термінальному пристрої, а $j \cdot C$ – складова, сформована циклічними властивостями експоненти, розглянутими вище. Загальна довжина в бітах для U_1, U_2 і W дорівнює H. У разі, якщо U_1 має найменше можливе значення, тобто $U_1 = 1$, то залишкова бітова довжина чисел U_2 і W становить H, а кількість можливих комбінацій їх значень дорівнює $(H+1) \cdot 2^{H-1}$. Якщо довжина U_1 дорівнює 2, тобто $h = 2$, і вона може мати значення 10 або 11, тоді залишкова довжина чисел U_2 і W дорівнює H-2, а отже, кількість варіантів їх значень становить $(H-1) \cdot 2^{H-3}$. З урахуванням двох можливих варіантів U_1, загальна кількість комбінацій для U_1, U_2 та W дорівнює $(H-1) \cdot 2^{H-2}$. Аналогічно, якщо довжина компоненти U_1 складає h бітів, то вона може мати 2^{h-1} варіантів, при цьому сума довжин U_2 і W дорівнює H-h, а кількість комбінацій для їх значень дорівнює $(H-h+1) \cdot 2^{H-h-1}$. Тоді загальна кількість

варіантів значень для трьох компонент становитиме $(H\text{-}h+1)\cdot 2^{H-2}$. Загальна кількість комбінацій V_2 для всіх можливих варіантів кодів U_1, U_2 і W, коли їх спільна довжина дорівнює H, обчислюється за формулою $V_2 = 2^{H-2}\cdot(H^2 + H)$. Таким чином, використання двокомпонентного мультиплікативно-адитивного розкладання виразу $K + j\cdot C = U_1\cdot F_1 + U_1\cdot F_2 + W$ дозволяє суттєво ускладнити процес перебору для зловмисника: кількість варіантів зростає в $(H^2 + H)/8$ разів порівняно з 2^H та в $H/4$ рази у порівнянні зі звичайним однокомпонентним мультиплікативно-адитивним розкладанням експоненти. Це безпосередньо підвищує стійкість до несанкціонованого відновлення секретного коду експоненти на сторонніх обчислювальних системах, які отримують лише часткову інформацію. Наприклад, якщо $H = 120$, то застосування описаного двокомпонентного розкладання дозволяє збільшити захищеність приблизно в 30 разів. За схожим принципом можна розрахувати й кількість можливих варіантів перебору при трьохкомпонентному мультиплікативно-адитивному розкладанні $K + j\cdot C = U_1\cdot F_1 + U_2\cdot F_2 + U_3\cdot F_3 + W$. Якщо U_1 дорівнює одиниці, то сумарна довжина компонент U_2, U_3 та W складає H, відповідно кількість комбінацій для цих трьох значень дорівнює $(H^2 + H)\cdot 2^{H-3} = H\cdot(H\text{-}1)\cdot 2^{H-3}$. Якщо ж довжина U_1 становить два біти ($h = 2$), тоді вона може набувати значень 10 або 11, а залишкова довжина для U_2, U_3 і W дорівнює H-2. Відповідно кількість комбінацій для них становить $(H\text{-}2)\cdot(H\text{-}3)\cdot 2^{H-5}$, а з урахуванням двох варіантів U_1 маємо загальну кількість варіантів $(H\text{-}2)\cdot(H\text{-}3)\cdot 2^{H-4}$. За аналогією, якщо розрядність U_1 дорівнює h, то кількість її можливих варіантів – 2^{h-1}, а довжина решти компонент (U_2, U_3, W) становить H-h. У цьому випадку кількість варіантів значень цих трьох компонент дорівнює $(H\text{-}h)\cdot(H\text{-}h\text{-}1)\cdot 2^{H-h-3}$, а загальна кількість варіантів значень для U_1, U_2, U_3 і W – це $(H\text{-}h)\cdot(H\text{-}h\text{-}1)\cdot 2^{H-4}$. Остаточне число всіх можливих комбінацій V_3, які відповідають варіантам перебору чотирьох чисел, що є кодами експонент для модулярного експоненціювання на термінальному мікроконтролері, обчислюється як:

$$V_3 = 2^{H-4} \cdot \sum_{h=2}^{H} h \cdot (h-1)\ . \qquad (3.21)$$

Сума, яка входить до складу формули (3.21), вбачається стандартною формулою цілочисельного ряду і обчислюється, згідно з [57], за формулою:

$$\sum_{h=2}^{H} h \cdot (h-1) = \frac{H \cdot (H+1) \cdot (H+2)}{3}\ . \qquad (3.22)$$

В такому випадку, якщо підставити формулу (3.22) у формулу (3.21), то можна отримати новий вигляд загальної кількості V_3 варіантів перебору для трьохкомпонентного мультиплікативно-адитивного розкладення:

$$V_3 = 2^{H-4} \cdot \sum_{h=2}^{H} h \cdot (h-1) = 2^{H-4} \cdot \frac{H \cdot (H+1) \cdot (H+2)}{3}\ . \qquad (3.23)$$

Отже, при застосуванні трьохкомпонентного мультиплікативно-адитивного розкладення $K + j \cdot C = U_1 \cdot F_1 + U_2 \cdot F_2 + U_3 \cdot F_3 + W$ об'єм перебору зловмисником значень U_1, U_2, U_3 и W для відновлення секретних розрядів коду експоненти збільшується в $H \cdot (H+2)/24$ разів, якщо порівнювати такий вид розкладення коду експоненти і традиційний, та у $(H+2)/6$ разів у порівнянні з двокомпонентним розкладенням коду у вигляді $K + j \cdot C = U_1 \cdot F_1 + U_2 \cdot F_2 + W$.

Як приклад ефективності підходу, можна навести ситуацію, коли при довжині коду експоненти H, що дорівнює 120 бітам, впровадження трьохкомпонентного мультиплікативно-адитивного розкладання забезпечує зростання рівня стійкості системи до несанкціонованої реконструкції секретного коду у 20 разів у порівнянні з варіантом, що використовує лише дві компоненти, та у 610 разів у порівнянні з традиційною схемою розкладання. Це свідчить про суттєве посилення криптографічного захисту навіть без зміни загальної довжини вхідного коду. Ще однією перевагою описаного підходу є можливість зменшити сумарну кількість бітів у тих частинах коду, що обробляються безпосередньо на стороні термінального пристрою – зокрема, мікроконтролера. Завдяки цьому досягається істотне зниження обчислювального навантаження на ресурсно обмежені пристрої, що є критично

важливим у контексті IoT, вбудованих систем і подібних застосувань. Проведені практичні експерименти підтвердили: використання багатокомпонентного розкладання дозволяє не тільки підтримувати заданий рівень захищеності, але й істотно підвищує швидкодію обчислень, пов’язаних із модулярним експоненціюванням, що часто є найвитратнішою операцією в криптографічних алгоритмах. Іншими словами, така методика забезпечує одночасно і підвищення безпеки, і покращення ефективності виконання обчислень, що робить її доцільною для широкого впровадження в захищених обчислювальних системах.

Висновки до розділу 3

У результаті досліджень, проведених в третьому розділі магістерської дисертації, спрямованих на підвищення рівня безпеки гомоморфного шифрування при реалізації модулярного експоненціювання, можна зробити наступні висновки.

1. Проведено аналіз перспективних шляхів для підвищення ефективності процесу модулярного експоненціювання на термінальних мікроконтролерах із залученням хмарних систем.

2. Теоретично обґрунтовано, розроблено та досліджено метод, що дозволяє досягти зростання рівня захищеності при виконанні модулярного експоненціювання на IoT-пристроях із залученням захищених хмарних обчислень. Запропонований метод ґрунтується на застосуванні багатокомпонентного розкладу експоненти, що поєднує мультиплікативні та адитивні складові. Це дозволяє динамічно збільшувати об’єм перебору, який потрібно здійснити зловмиснику для несанкціонованої рекунструкції секретного параметру модулярного експоненціювання – закритого ключа. Кількість компонент розкладення визначається конкретним практичним застосуванням розробленого методу. Зокрема розроблений метод рекомендовано використовувати в реальному часі для систем віддаленого моніторингу, що функціонують на базі IoT-технологій. Концепція методу суттєво ускладнює процес зловмисної реконструкції секретних ключа за даними, що передаються на віддалену систему, тим самим підвищуючи стійкість обчислень базової для криптографії з відкритим ключем операції – модулярного експоненціювання.

3. Проведений аналіз ефективності запропонованого методу свідчить про значне зростання рівня криптографічного захисту, що забезпечується запропонованим методом. Розроблена в рамках розділу математична модель демонструє, що збільшення кількості складових у багатокомпонентному мультиплікативно-адитивному розкладенні експоненти прямо впливає на

підвищення рівня захищеності. Показано, що зокрема при довжині коду експоненти, що дорівнює 120 бітам, впровадження трьохкомпонентного мультиплікативно-адитивного розкладання забезпечує зростання рівня стійкості системи до несанкціонованої реконструкції секретного коду у 20 разів у порівнянні з варіантом, що використовує лише дві компоненти, та у 610 разів у порівнянні з традиційною схемою розкладання.

РОЗДІЛ 4. РОЗРОБКА ПРОГРАМНИХ ЗАСОБІВ МОДЕЛЮВАННЯ МЕТОДУ БЕЗПЕЧНОГО ЗАЛУЧЕННЯ ХМАРНИХ СИСТЕМ

4.1 Обґрунтування вибору структури даних в рамках програмної реалізації

Програмна реалізація призначена продемонструвати реалізацію операції модулярного експоненціювання у вигляді розподілених обчислень між термінальним пристроєм та віддаленої системою. Роль термінального пристрою у програмі виконує клієнт, в той час, як у ролі віддаленої системи виступає сервер. Основна структура програми включає в себе три пакети: longintegerarithmetics, modularoperations та secureexponentiation, що відповідають базовим рівням обробки даних, операціям модулярної арифметики та організації захищеного клієнт-серверного протоколу відповідно.

У процесі реалізації програмного забезпечення було використано декілька специфічних структур даних, зокрема:

- об’єкт для подання довгих додатних цілих чисел довільної розрядності (структура LongInteger);
- структура, що зберігає зашифровані значення основи та експоненти перед їх передачею для обчислення на віддаленому сервері (BlindingData);
- об’єкт для подання результату операції ділення у вигляді частки та остачі (DivisionResult).

Основна увага в цьому розділі приділяється саме структурі LongInteger, яка відіграє ключову роль у функціонуванні всієї системи, оскільки є універсальним представленням довгих чисел, з якими виконуються всі основні арифметичні та логічні операції. Інші допоміжні структури – BlindingData та DivisionResult – розглядаються окремо у контексті відповідних програмних модулів, до яких вони належать. Клас LongInteger реалізовано на основі масиву 32-бітних цілих чисел, що виступає в ролі внутрішнього сховища числових даних. Для роботи з цим масивом реалізовано набір спеціалізованих методів.

Зокрема, метод getValue() повертає повний масив числових блоків. Методи getInt(int) та getLong(int) дозволяють отримати значення окремого елемента масиву у 32-бітному та 64-бітному форматі відповідно. Аналогічно, setInt(int) та setLong(int) забезпечують запис значень у відповідних форматах. Через те, що мова Java не підтримує беззнакові типи, збереження точності значень при роботі з великими числами забезпечується використанням 64-бітних представлень. Якщо звернення до масиву відбувається з індексом, що перевищує поточну довжину масиву, метод getInt(int) або getLong(int) повертає нуль. У разі, коли потрібно записати значення за індексом, який перевищує розмір масиву, внутрішня структура автоматично розширюється, щоб забезпечити відповідну ємність. Поточну кількість 32-бітних елементів, які складають число, можна отримати за допомогою методу getSize(). Метод stripLeadingZeroes() забезпечує очищення старших (найлівіших) розрядів числа від надлишкових нулів, оптимізуючи представлення числа та зменшуючи його об’єм. Для реалізації бітових зсувів [58] передбачено методи shiftLeft() та shiftRight(), які зсувають значення на один біт відповідно вліво або вправо. Існують також перевантажені версії цих методів, що дозволяють вказати кількість бітів для зсуву (shiftLeft(int) та shiftRight(int)). Після зсуву загальний розмір числа залишається незмінним, проте крайні біти можуть бути втрачені.
Порівняння об’єктів типу LongInteger реалізовано через метод compareTo(LongInteger). Він дозволяє встановити, чи є два числа рівними, а також визначити, яке з них більше або менше. У випадку рівності повертається нульове значення, для меншого – від’ємне, а для більшого – додатне.

4.2 Розробка програмної реалізації

Розроблена програма складається з чотирьох основних пакетів: longintegerarithmetics (відповідає за базові арифметичні обчислення над числами довільної довжини), modularoperations (реалізує модулярні операції), secureexponentiation (містить логіку захищеного модулярного

експоненціювання, включно з обробкою на клієнті та сервері), а також допоміжного пакета utils.

Пакет longintegerarithmetics включає модульні підсистеми для реалізації чотирьох основних арифметичних операцій: додавання (adder), віднімання (subtractor), множення (multiplier) та ділення (divisor). Кожна з них представлена через інтерфейс та відповідну реалізацію. Арифметичні операції над об'єктами типу LongInteger здійснюються в основі 2^{32}, що забезпечує ефективну роботу з дуже великими числами. Наприклад, додавання та множення реалізовано класичними алгоритмами, які враховують розмірність операндів, а ділення базується на адаптованому алгоритмі Кнута і повертає як частку, так і залишок у вигляді об'єкта DivisionResult.

Пакет modularoperations відповідає за реалізацію модулярних арифметичних операцій. Його структура поділена на три частини: modularcalculator, modularmultiplier та modularexponentiator. У першому реалізовано знаходження залишку від ділення з використанням ділення великих чисел (Divisor). У другому – модулярне множення з редукцією результату на кожному кроці [59]. У третьому містяться три алгоритми для модулярного експоненціювання: класичний двійковий зсув справа наліво (BinaryRightToLeftModularExponentiator), алгоритм Монтгомері [60] (MontgomeryExponentiator) та реалізація віддаленого експоненціювання через HTTP-запити (RemoteExponentiator).

Пакет secureexponentiation забезпечує логіку захищеного виконання модулярного експоненціювання з розподілом обчислень між локальним клієнтом і віддаленим сервером. У його складі виокремлюються підпакети blinding, client та server.

- У blinding реалізовано механізми затемнення вхідних даних – основи та експоненти – з метою забезпечення конфіденційності. Генерація здійснюється через BlindingDataProviderImpl, що формує структуру

BlindingData, яка гарантує математичну коректність модифікованих значень відповідно до заданих умов.

- Пакет client містить класи Client, ClientConfig і TestClient, які відповідають за ініціалізацію, налаштування та виклик обчислювального процесу з боку користувача. Клас Client є універсальним викликачем, який координує обчислення, використовуючи реалізацію SecureExponentiator. Конфігурація клієнта (ClientConfig) базується на Spring-анотаціях та задає компоненти за замовчуванням, зокрема, алгоритм Монтгомері для локальних обчислень і REST-запити до сервера для віддалених.
- У новому класі StepController, також включеному до клієнтської частини, реалізовано можливість поетапного виконання обчислень. Цей клас контролює кожен крок алгоритму, розбиваючи загальний процес експоненціювання згідно з обраною схемою. Такий підхід дозволяє гнучко керувати порядком обчислень, частково виконувати їх локально, а частково делегувати серверу.
- Пакет server відповідає за обробку REST-запитів, що надходять від клієнта. Він перетворює вхідні параметри з шістнадцяткового текстового представлення у формат LongInteger, виконує обчислення за допомогою вибраного алгоритму експоненціювання (наприклад, Монтгомері [61]), та повертає результат у вигляді рядка. Серверна частина побудована на Spring Boot з використанням Tomcat як вбудованого сервлет-контейнера. Для реалізації модулярного експоненціювання на сервері за замовчуванням використовується MontgomeryExponentiator.

Інфраструктура побудована на Maven, із підключенням бібліотек spring-boot-starter та spring-boot-starter-web, що спрощують налаштування клієнт-серверної взаємодії. Взаємодія між клієнтом і сервером ґрунтується на обміні повідомленнями через HTTP з кодуванням даних у шістнадцятковій формі. Структура компонентів дозволяє розширення системи, наприклад, через

додавання нових алгоритмів експоненціювання або альтернативних схем захисту.

Для запуску реалізованого програмного забезпечення захищеного модулярного експоненціювання на довгих цілих числах необхідне попереднє забезпечення відповідного програмного середовища. Основними системними вимогами є наявність встановленого комплекту Java Development Kit (JDK) версії 17 або вище, системи керування залежностями Maven (рекомендована версія – не нижче 3.6), а також сучасної операційної системи, що підтримує виконання Java-програм, зокрема Windows, macOS або Linux. Для збирання та виконання проєкту бажано мати не менше 2 ГБ оперативної пам’яті, а також приблизно 100 МБ вільного місця на диску для зберігання зібраного коду та зовнішніх бібліотек.

Проєкт реалізовано у вигляді багатомодульної Java-програми, що включає окремі компоненти для клієнтської та серверної частини, які взаємодіють у процесі віддаленого захищеного модулярного експоненціювання. Основні обчислювальні модулі розташовані в окремих пакетах, які відповідають за базові арифметичні операції, модулярне множення, обчислення степеня за модулем, а також за процедури блайндінгу. Центральними для запуску є компоненти, що реалізують клієнтський (пакет secureexponentiation.client) та серверний (пакет secureexponentiation.server) функціонал.

Для ініціалізації роботи програми користувач повинен клонувати репозиторій із проєктом за допомогою системи контролю версій (наприклад, Git) і здійснити збірку проєкту через команду mvn clean install. Після завершення збирання необхідно запустити серверну частину, яка реалізована на основі фреймворку Spring Boot. Це досягається виконанням класу з головним методом у пакеті server, після чого сервер буде очікувати вхідні HTTP-запити на стандартному порті (наприклад, 8080). Клієнтська частина викликається шляхом запуску класу TestClient, виконує blinding, розраховує проміжні

значення, ініціює запит до сервера для обчислення віддаленого степеня, отримує результат та виконує зворотні обчислення для відновлення підсумкового результату $D^K \bmod M$. Зміна реалізації алгоритмів (наприклад, вибір між методом Монтгомері [62] або правостороннім методом) здійснюється через відповідні конфігураційні файли або шляхом зміни реалізацій інтерфейсів у класі конфігурації клієнта. Також у програмі реалізовано систему модульності, що дозволяє розширювати проєкт, підключаючи нові арифметичні модулі або методи блайндінгу без порушення цілісності основної логіки. Для перевірки коректності реалізації у проєкті передбачено набір юніт-тестів, які охоплюють основні сценарії роботи алгоритму. Їх виконання здійснюється командою mvn test.

4.3 Результати програмної реалізації

Розроблена в рамках розділу програмна реалізація спрямована на моделювання практичного застосування методу безпечного залучення хмарних систем для прискореної реалізації операцій несиметричної криптографії. Теоретично ефективність запропонованого методу показується на такому критерії, як рівень захищеності, що в свою чергу визначається об’ємом перебору V_k, який необхідно здійснити зловмиснику для несанкціонованого відновлення закритого ключа експоненти. Індекс k показує рівень багатокомпонентного мультиплікативно-адитивного розкладення за розробленим методом. Теоретична оцінка об’єму перебору V_k розробленого методу показала експоненційне зростання рівня захищеності обчислень із зростанням кількості компонент мультиплікативно-адитивного розкладення коду експоненти. Фактично графік залежності рівня захищеності від кількості компонент розкладення в рамках теоретичного дослідження наведені на рис. 4.1:

Рис. 4.1. Теоретична залежність рівня захищеності від кількості компонент мультиплікативно-адитивного розкладення коду експоненти

Практичні дослідження рівня захищеності розробленого методу безпечного залучення хмарних систем, проведені в рамках програмної реалізації, були спрямовані на визначення реального об'єму перебору N_k, потрібного для реконструкції секретного ключа модулярного експоненціювання. Експерименти підтвердили загальну тенденцію експоненційного зростання рівня захищеності зі збільшенням кількості компонент мультиплікативно-адитивного розкладення коду експоненти. Реальні результати програмного визначення об'єму перебору для зламу експоненти показали близькі, хоча дещо гірші за теоретичні розрахунки, результати. Ці результати також доцільно представити у вигляді графіку, наведеному на рис. 4.2:

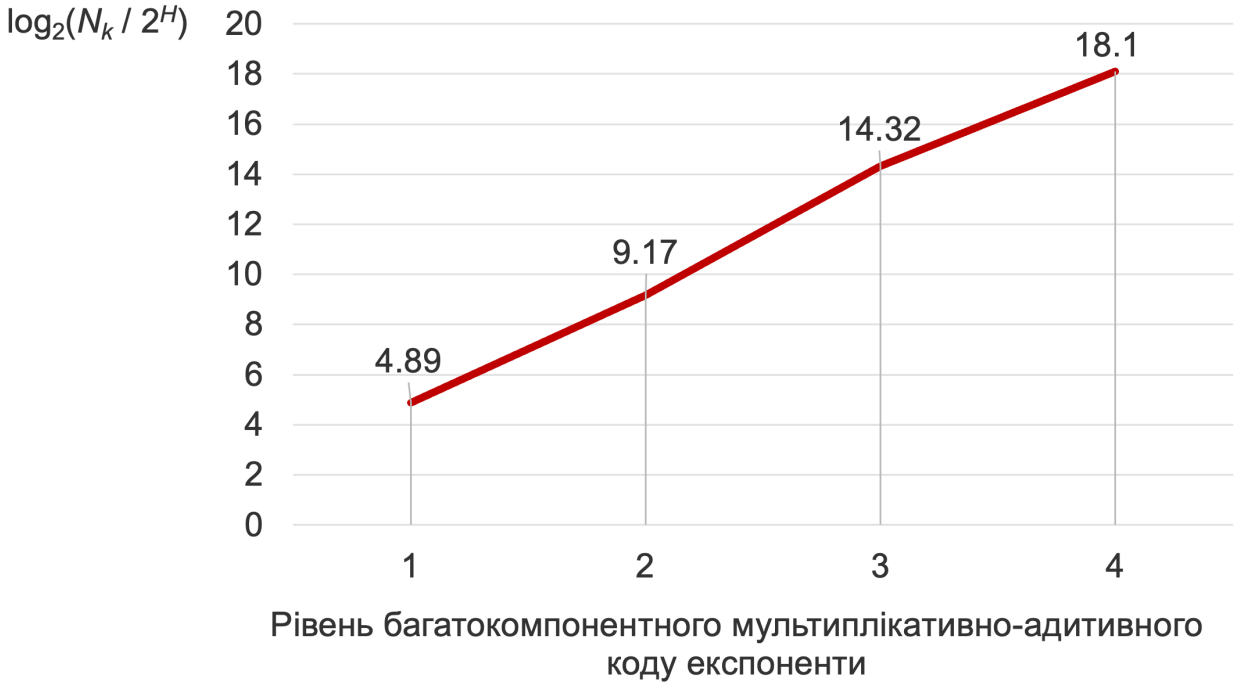

Рис. 4.2. Практична залежність рівня захищеності від кількості компонент мультиплікативно-адитивного розкладення коду експоненти

Висновки до розділу 4

В ході виконання четвертого розділу магістерської дисертації, присвяченого розробці та дослідженню програмного забезпечення для захищеного обчислення модулярної експоненти за розробленим методом безпечного залучення віддалених систем для прискореної реалізації несиметричної криптографії, отримано наступні результати:

1. Розроблено спеціалізоване програмне забезпечення, яке імітує основні етапи обчислення модулярної експоненти за запропонованим методом захищеного модулярного експоненціювання у віддалених комп'ютерних системах, для отримання об'єктивних та достовірних оцінок ефективності розробленого методу. Розроблене програмне забезпечення забезпечує повну підтримку математичної моделі, описаної в третьому розділі магістерської дисертації, що дозволило провести експериментальне підтвердження коректності розробленого методу.
2. Перевірено, що результати програмного обчислення модулярної експоненти за запропонованим методом ідентичні результатам теоретичного обчислення, що підтверджує правильність програмної реалізації.
3. Експериментальні результати засвідчили, що впровадження розробленого методу дозволяє суттєво підвищити рівень захищеності обчислення модулярної експоненти в системах віддаленого керування за рахунок збільшення кількості компонент мультиплікативно-адитивного розкладення коду експоненти.

ВИСНОВКИ

В результаті проведених досліджень в рамках магістерської дисертації, які спрямовані на огляд та дослідження задачі прискорення захищеної реалізації операцій несиметричної криптографії у хмарах, а також на розробку методів вирішення цієї задачі можуть бути зроблені наступні висновки:

1. Проведений аналіз практичного використання криптографії з відкритим ключем показав, що найбільш критичним вбачається застосування цих технологій в системах віддаленого моніторингу та керування об'єктами реального світу. При цьому основним чинником виступає обмеженість обчислювальних ресурсів термінальних комп'ютерних платформ, які часто не можуть забезпечити реалізацію функцій контролю автентичності даних та команд в режимі реального часу. З огляду на те, що термінальні мікроконтролери оснащені радіомодемом, одним з найбільш перспективних шляхів підвищення ефективності реалізації криптографічних протоколів з відкритим ключем на цих пристроях полягає у залученні до обчислень віддалених комп'ютерних систем.

2. В результаті аналізу відомих технологій залучення хмарних систем виявлено два можливих варіанти організації розподілених обчислень модулярної експоненти. Перший з них передбачає наявність на віддаленій платформі спеціалізованого криптопроцесора, обчислювальна потужність якого дозволяє реалізувати модулярне експоненціювання на порядки швидше, ніж на термінальному мікроконтролері. Водночас, застосування криптопроцесора на віддаленій системі не дозволяє обмінюватись проміжними результатами обчислень з мікроконтролером. Другий спосіб організації розподілених обчислень полягає у використанні в хмарі високопотужних процесорних засобів без залучення криптопроцесорів. Такий варіант організації передбачає можливість обміну проміжними результатами обчислень з мікроконтролером. Проведені дослідження показали, що більш доцільним є реалізація саме другого способу розподілу обчислень модулярної експонети.

3. Детальний аналіз алгоритмів модулярного експоненціювання показав, що більші перспективи для розподіленого обчислення надає алгоритм експоненціювання з молодших розрядів, який, фактично, містить дві гілки обчислень, одна з яких може вважатися несекретною: а саме n операцій послідовного піднесення до квадрату числа, над яким виконується експоненціювання. Ця гілка може повністю виконуватися на віддаленій комп'ютерній платформі. Обчислення другої гілки частково або повністю можуть бути реалізовані на термінальній обчислювальній платформі з урахуванням результатів віддалених обчислень. Потрібно дослідити ефективні форми такої взаємодії для розробки ефективних методів організації розподілених обчислень.

4. На теоретичному рівні обґрунтовано, що одним із ключових шляхів підвищення ефективності є зменшення обчислювального навантаження на обмежені в ресурсах термінальні мікроконтролери. Такий підхід дозволяє суттєво скоротити загальний час виконання операції за рахунок делегування складних обчислень на потужні віддалені ресурси.

5. Розглянуто та проаналізовано можливі шляхи вирішення проблеми підвищення ефективності реалізації модулярного експоненціювання на термінальних пристроях Інтернету Речей.

6. Теоретично обґрунтовано, розроблено та досліджено метод реалізації модулярного експоненціювання, що базується на безпечному розподілі обчислень між термінальним пристроєм та хмарною інфраструктурою. Суть методу полягає в тому, що замість послідовного виконання модулярних множень на термінальному мікроконтролері відповідно до розрядів експоненти, система попередньо обчислює всі можливі варіанти на хмарній системі. Після цього термінальний пристрій лише обирає необхідні результати згідно зі своїм ключем. Це дозволяє значно скоротити кількість обчислень, які мають бути виконані безпосередньо на пристрої користувача, що, в свою чергу, призводить до пришвидшення всієї процедури обчислення експоненти.

7. Розроблено математичну модель процесу такого розподіленого обчислення, яка дозволяє раціонально підібрати ключові параметри методу з урахуванням характеристик як термінального пристрою, так і віддаленого середовища.

8. Результати теоретичних розрахунків та проведених експериментів узгоджуються між собою та свідчать про значне підвищення продуктивності. Зокрема, реалізація запропонованого методу забезпечує прискорення виконання операції модулярного експоненціювання в середньому в 3–4 рази порівняно з існуючими методами, що також передбачають залучення хмарних обчислень. Це підтверджує доцільність застосування розробленого підходу в практичних системах криптографічного захисту для IoT-середовищ, де ресурси термінальних пристроїв є обмеженими.

9. Проведено аналіз перспективних шляхів для підвищення ефективності процесу модулярного експоненціювання на термінальних мікроконтролерах із залученням хмарних систем.

10. Теоретично обґрунтовано, розроблено та досліджено метод, що дозволяє досягти зростання рівня захищеності при виконанні модулярного експоненціювання на IoT-пристроях із залученням захищених хмарних обчислень. Запропонований метод ґрунтується на застосуванні багатокомпонентного розкладу експоненти, що поєднує мультиплікативні та адитивні складові. Це дозволяє динамічно збільшувати об’єм перебору, який потрібно здійснити зловмиснику для несанкціонованої рекунструкції секретного параметру модулярного експоненціювання – закритого ключа. Кількість компонент розкладення визначається конкретним практичним застосуванням розробленого методу. Зокрема розроблений метод рекомендовано використовувати в реальному часі для систем віддаленого моніторингу, що функціонують на базі IoT-технологій. Концепція методу суттєво ускладнює процес зловмисної реконструкції секретних ключа за даними, що передаються на віддалену систему, тим самим підвищуючи

стійкість обчислень базової для криптографії з відкритим ключем операції – модулярного експоненціювання.

11. Проведений аналіз ефективності запропонованого методу свідчить про значне зростання рівня криптографічного захисту, що забезпечується запропонованим методом. Розроблена в рамках розділу математична модель демонструє, що збільшення кількості складових у багатокомпонентному мультиплікативно-адитивному розкладенні експоненти прямо впливає на підвищення рівня захищеності. Показано, що зокрема при довжині коду експоненти, що дорівнює 120 бітам, впровадження трьохкомпонентного мультиплікативно-адитивного розкладання забезпечує зростання рівня стійкості системи до несанкціонованої реконструкції секретного коду у 20 разів у порівнянні з варіантом, що використовує лише дві компоненти, та у 610 разів у порівнянні з традиційною схемою розкладання.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Русанова О.В. Метод безпечного розподіленого обчислення модулярної експоненти для прискорення реалізації механізмів захисту даних в IoT / О.В. Русанова, М.А. Гайдукевич, Міратанаї Аліреза // Проблеми управління та інформатизації.- 2023.- № 4 (77).- С.74-87. DOI: 10.18372/2073-4751.76.18243
2. Kumar S. Internet of Things is a revolutionary approach for future technology enhancement: review / S. Kumar, P. Tiwari, M. Zymbler // Journal of Big Data. – 2019. – № 6. – С. 63–71.
3. Elgazzar K. Revisiting the internet of things: New trends, opportunities and grand challenges / K. Elgazzar, H. Khalil, T. Alghamdi, A. Badr, G. Abdelkader Abdelrahman Elewah, R. Buyya // Frontiers in Internet of Things. – 2022. – Vol. 1. – С. 1–18. https://doi.org/10.3389/friot2022.1073780.
4. Jurcut A.D. Introduction to IoT Security / A.D. Jurcut, R.R. Xu // IoT Security: Advances in Authentication. – 2020. – С. 1–64. https://doi.org/10.1002.9781119527978.ch2.
5. Unal D. A secure and efficient Internet of Things cloud encryption scheme with forensics investigation compatibility based on identity-based encryption / D. Unal, A.A. Ali-Ali, F.O. Catak // Future Generation Computer Systems. – 2021. – Vol. 125. – С. 433–445. https://doi.org/10.1016/J.FUTURE.2021.06.050.
6. Meneghello F. IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices / F. Meneghello, M. Calore, D. Zucchetto, M. Polese, A. Zalella // IEEE Internet of Things Journal. – 2019. – Vol. 6, № 5. – С. 8182–8201. https://doi.org/10.1109/JIOT.2019.2935189.
7. Van Dijk M. Fully homomorphic encryption over the integers / M. Van Dijk, C. Gentry, S. Halevi, V. Vaikuntanathan // Annual International Conference on the Theory and Applications of Cryptographic Techniques. – 2010. – С. 24–43.
8. Menezes A. Handbook of Applied Cryptography / A. Menezes, P.C. van Oorschot, S.A. Vanstone. – CRC Press, 2001. – 780 с.

9. Mirataei A. Fast secure calculation of the open key cryptography procedures for IoT in clouds / A. Mirataei, M. Haidukevych, O. Markovskyi // Information, Computing and Intelligent Systems. – 2022. – № 3. – С. 56–62.

10. National Institute of Standards and Technology. Announcing the Advanced Encryption Standard (AES) / U.S. Department of Commerce. – Gaithersburg, MD: NIST, 2001. – 51 с.

11. Dhem J.-F., Quisquater J.-J., “Resent results on modular multiplications for smart cards”. Proceedings of GARDIS 1998. LNCS-1820, Springer-Verlag, 2000, pp. 350-366.

12. Hasenplaugh, W.; Gaubatz, G. Gopal, V., “Fast Modular Reduction Computer Arithmetic”, ARITH '07. 18th IEEE Symposium on 25-27 June 2007 Page(s):225 - 229

13. Jaewook Chung; Anwar Hasan, M. b; “Montgomery Reduction Algorithm for Modular Multiplication Using Low-Weight Polynomial Form Integers”, Computer Arithmetic, 2007. ARITH '07. 18th IEEE Symposium on 25-27 June 2007. Page(s):230 – 239

14. Noot M.M. Current research on Internet of Things (IoT) / M.M. Noot, W.H. Hassan // Compute Network. – 2019. – Vol. 148, № 15. – С. 283–294.

15. Paar C. Understanding Cryptography: A Textbook for Students and Practitioners / C. Paar, J. Pelzl. – Berlin: Springer, 2010. – 372 с.

16. Ding Y. A High-Performance RSA Coprocessor Based on Half-Carry-Save and Dual-Core MAC Architecture / Y. Ding, J. Hu, D. Wang, H. Tan // Chinese Journal of Electronics. – 2018. – Vol. 27, № 1. – С. 1–6.

17. Hussein, H. I. An efficient ElGamal cryptosystem scheme / Hussein, H. I., Abdullah, W. M. // *International Journal of Computers and Applications*. – 2021. – Vol. 43, No. 10. – P. 1041–1047. – DOI: 10.1080/1206212X.2019.1678799.

18. Saepulrohman, A. Data integrity and security of digital signatures on electronic systems using the digital signature algorithm (DSA) / Saepulrohman, A., Ismangil, A.

// *International Journal of Electronics and Communications Systems*. – 2021. – Vol. 7, No. 2. – P. 144–151. – [Електронний ресурс]. – Режим доступу: https://ejournal.radenintan.ac.id/index.php/IJECS/article/view/7923, вільний. – Назва з екрана.

19. Ruiz A.L. Two Galois Fields Cryptographic Applications / A.L. Ruiz, E. Castillo, L. Parrilla, A. Garsia // Aritmetic and Algebraic Circuits.- 2021.- P.551-565. DOI: 10. 1007/978-3-030-67266-9_12

20. Zholubak M. Galua Field Multipliers Core Generator / M. Zholubak, V. S. Hlukhov // International Journal of Computer Network and Information Security(IJCNIS), - 2023.- Vol.15,- № 3.-P.1-14. DOI:10.5815/ ijcnis. 2023.03.01.

21. Boiarshyn A. Organization of parallel execution of modular multiplication to speed up the computational implementation of public-key cryptography / A. Boiarshyn, O. Markovskyi, B. Ostrovska // Information, Computing and Intelligent Systems. – 2022. – № 3. – С. 26–32.

22. Markovskyi O.P. Secure Modular Exponentiation in Cloud Systems / O.P. Markovskyi, N. Bardis, N. Doukas, S. Kirilenko // Proceedings of The Congress on Information Technology, Computational and Experimental Physics (CITCEP 2015). – 2015. – С. 266–269.

23. Khan M.A. IoT security: Review, blockchain solution and open challenges / M.A. Khan, K. Salah // Future Generation Computer Systems. – 2018. – Vol. 82, № 5. – С. 395–411.

24. Hong S.M. New modular multiplication algorithms for fast modular exponentiation / S.M. Hong, S.Y. Oh, H. Yoon // Proceeding of Advances in Cryptology Eurocrypt’96. – Springer-Verlag, 1996. – С. 166–177.

25. Русанова О.В. Метод розподіленого модулярного експоненціювання на термінальних мікроконтролерах IoT із захищеним залученням хмарних обчислень / О.В. Русанова, М.А. Гайдукевич // Проблеми автоматизації та управління. – 2024. – Т. 2, № 78. – С. 91–103.

26. Fox G. Using Clouds for Technical Computing / G. Fox, D. Gannon // Cloud Computing and Big Data. – Amsterdam: IOS Press, 2018. – С. 81–102.
27. Tselios C. Enhancing SDN security for IoT-related deployments through blockchain / C. Tselios, I. Politis, S. Kotsopoulos // 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN). – Berlin, 2017. – С. 303–308.
28. Haches G. Montgomery multiplication with no final subtraction / G. Haches, J.J. Quisquater // Cryptographic Hardware and Embedded Systems – CHES'2013. – Springer-Verlag, 2013. – С. 293–301.
29. Boyko V. Speeding up log and factorization-based schemes via precomputations / V. Boyko, M. Pienado, R. Venkatesan // Advances in Cryptography – Proceeding of EUROCRYPT'98. – Springer-Verlag, 1998. – С. 221–235.
30. Марковський О.П. Захищена реалізація фільтрації зображень в GRID-системах / О.П. Марковський, М.В. Невдащенко, А.М. Білашевська // Вісник Національного технічного університету України "КПІ". Інформатика, управління та обчислювальна техніка. – Київ: ВЕК+, 2014. – № 61. – С. 105–109.
31. Bardis N. Secure Implementation of Modular Exponentiation on Cloud Computing Resources / N. Bardis, O. Markovskyi // Proceedings of International Conference Applied Mathematics, Computational Science and Systems Engineering. – Athens, Greece, 2017. – С. 90–96.
32. Borges J.A. A Secure Cloud Computing Method for Rapid Implementation of Cryptographic Data Protection in IoT / J.A. Borges, O. Haidukevych, A. Mirataei, N. Doukas // Proceedings of The 13th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT'2023. – Athens, Greece, 2023. – С. 50–53.
33. Bardis N. Secure, Green Implementation of Modular Arithmetic Operations for IoT and Cloud Applications / N. Bardis // Green IT Engineering: Components,

Networks and System Implementation. – 2017. – С. 43–64. DOI: 10.1007/978-3-319-55595-9_3.

34. Марковський О.П. Метод прискорення експоненціювання з використанням передобчислень / О.П. Марковський, О.В. Русанова, А.А. Олієвський, В.М. Черевик // Телекомунікаційні та інформаційні технології. – 2018. – № 1(58). – С. 31–39.

35. Kawamura S. A fast modular exponentiation algorithm / S. Kawamura, K. Takabayashi, A. Shimbo // IEEE Transactions on Information Theory. – 2015. – Vol. 94. – № 6. – P. 2136–2142.

36. Bos J.N. Additional chain heuristics / J.N. Bos, M. Coster // Cryptographic Hardware and Embedded Systems – CHES'2014. – Springer-Verlag, 2014. – С. 143–151.

37. Романець Ю.А. Захист інформації в комп'ютерних системах і мережах / Ю.А. Романець, П.А. Тимофєєв, В.Ф. Шаньгін. – М.: Радіо і зв'язок. – 2001. – 376 с.

38. Stinson D. R. Cryptography: Theory and Practice / D. R. Stinson, M. B. Paterson. – 4th ed. – Boca Raton: CRC Press. – 2019. – 612 p.

39. P. Lara, F. Oliveira, and R. Portugal, "Paralelizao eficiente para o algoritmo binrio de exponenciao modular," in IX Simpsio Brasileiro Segurana da Informao e Sistemas Computacionais. SBC, 2009, pp. 17–26.

40. Katz J. Introduction to Modern Cryptography / J. Katz, Y. Lindell. – 3rd ed. – Boca Raton: CRC Press. – 2020. – 603 p.

41. Boroujerdi N. Cloud Computing: Changing Cogitation about Computing / N. Boroujerdi, S. Nazem // IJCSI International Journal of Computer Science Issues. – 2012. – Vol. 9, Issue 4, No. 3. – С. 169–180.

42. Задірака В.К. Комп'ютерна арифметика багато розрядних чисел: Наукове видання / В.К. Задірака, О.С. Олексик. – К.: Вища школа, 2003. – 264 с.

43. International Business Machines Corporation. Hardware Cryptographic Accelerators in Embedded Systems / IBM Redbooks. – Boca Raton, FL: IBM Corporation, 2021. – 76 с.
44. SafeLogic. CryptoSwift 600 Technical Datasheet [Електронний ресурс]. – Режим доступу: https://www.safelogic.com/cryptoswift600 – Назва з екрана.

45. Kocher P. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems / P. Kocher // Advances in Cryptology – CRYPTO'96. – 1996. – Vol. 1109. – C. 104–113.

46. Hodjat A. A 21.54 Gbps Fully Pipelined AES Processor on FPGA / A. Hodjat, I. Verbauwhede // 12th IEEE Symposium on Field-Programmable Custom Computing Machines, 2004. – C. 308–309.

47. Chen J. The impact of broadband speed on innovation: City-level evidence from China / J. Chen, J. Wang // Heliyon. – 2023. – Vol. 9, No. 1. – e12692.

48. Zuras D., More on squaring and multiplying larges integers, IEEE Transactions on Computers,-1994.- Vol. 43, - № 8,- P. 899–908.
49. Barrett P. Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor // Proceedings CRYPTO'86. – P. 311-323.
50. Toom A. L., The complexity of a scheme of functional element realizing the multiplication of integers. Soviet Mathematics, vol. 3, pp. 714–716, 1963.
51. Markovskyi O. P., Ababne M.M.A. A method for accelerated implementation of modular exponentiation with a constant modulus. Visnyk of Natinal Technical University of Ukraine KPI. Informatics, control and computer technic. – 2007.- № 46.- pp. 115-122.

52. N.Doukas, A.Drigas, NG.Bardis, N.V.Karadimas, Accessible Secure Information Society Applications via the Use of Optimised Cryptographic

Calculations, Journal of Applied Mathematics & Bioinformatics, vol.3, no.4, 2013, 181-206, ISSN: 1792-6602,Scienpress Ltd, 2013.

53. Buhrow B., Gilbert B., Haider C. Parallel modular multiplication using 512-bit advanced vector instructions: RSA fault-injection countermeasure via interleaved parallel multiplication. Journal of Cryptographic Engineering.-2021.- № 2.- https://doi.org/10.1007/s13389-021-00256-9

54. Laszlo Hars.. “Long Modular multiplication for Cryptographic Applications”, Cryptographic Hardware and Embedded System- CHES’2004. LNCS-3156, Springer-Verlag, 2004, pp.45-61.

55. Xiaofeng Chen1, Jin Li, Jianfeng Ma3, Qiang Tang, Wenjing Lou. New Algorithms for Secure Outsourcing of Modular Exponentiations. ESORICS 2012, LNCS 7459, - 2012.- PP. 541–556.

56. Marcelo E. Kaihara and Naofumi Takagi, “Bipartite Modular Multiplication Method”; IEEE Transactions on Computers, Vol. 57, NO. 2, February 2008.

57. Gradshteyn I.S. Table of Integrals, Series and Products / I.S. Gradshteyn, I.M. Ryzhik. – 7th ed. – London: Academic Press, 2011. – 1176 c.

58. Che Wun Chion, Ted C.Yang. Parallel modular multiplication with table look-up. International Journal of Computer Mathematics.-1998.-Vol.69.-Issue 1-2.- P.22-23. https://doi.org/10.1080/00207189808804708

59. Montgomery P. Modular multiplication without trial division. // Mathematics of Computation. – 44(170). – 1985. – P. 519–521.

60. Kaya Koc. Analyzing and comparing Montgomery multiplication algorithms. / Kaya Koc, C., Acar, T., Kaliski, B.S. // IEEE Micro 16(3), 26–33 (1996). https://doi.org/10.1109/40.502403 15.

61. Bos. Montgomery multiplication using vector instructions / Bos, J.W., Montgomery, P.L., Shumow, D., Zaverucha, G.M. // Selected Areas in Cryptography–SAC, August 14–16, 2013, pp. 471–489. https://doi.org/10.1007/978-

3-662-43414-7_24

62. Bernstein D. J. Algorithmic Number Theory. MSRI Publications, 2008, vol. 44, ch. Fast multiplication and its applications, pp. 325–384.