НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

«На правах рукопису»

УДК 003.26.09:519.2

«До захисту допущено»
Завідувач кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2024 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою
«Математичні методи криптографічного захисту інформації»

зі спеціальності: 113 Прикладна математика
на тему: **«Криптографічні атаки на AES на основі інформації з побічного каналу»**

Виконав:
студент II курсу, групи ФІ-84
Толмачов Євгеній Юрійович _____

Керівник:
доцент, д.ф.-м.н., професор
Савчук Михайло Миколайович _____

Рецензент:
професор, д.ф.-м.н., зав. кафедри
Клесов Олег Іванович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ — 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

Рівень вищої освіти — другий (магістерський)
Спеціальність — 113 Прикладна математика,
ОПП «Математичні методи криптографічного захисту інформації»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2024 р.

ЗАВДАННЯ
на магістерську дисертацію

Студент: Толмачов Євгеній Юрійович

1. Тема роботи: «*Криптографічні атаки на AES на основі інформації з побічного каналу*», науковий керівник дисертації: доцент, д.ф.-м.н., професор Савчук Михайло Миколайович,

затверджені наказом по університету №__ від «__» _____ 2024 р.

2. Термін подання студентом роботи: «__» _____ 2024 р.

3. Об'єкт дослідження: *системи криптографічного захисту інформації*

4. Предмет дослідження: *алгоритми виправлення помилок в отриманому криптоаналітиком ключі*

5. Перелік завдань:

– *Побудова моделей спотворення ключа;*

– *Побудода атаки виправлення помилок в спотворенному ключі;*

– *Оцінка ймовірність успіху та складність атак для отриманих моделей;*

– *Програмна реалізація обрахунку оцінки ймовірності успіху та складності атак для отриманих моделей.*

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: Презентація доповіді

7. Орієнтовний перелік публікацій: *XXI Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» (Україна, м. Київ, 17 Травня 2024 р.)*

8. Дата видачі завдання: 10 вересня 2023 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2023 р.	Виконано
2	Ознайомлення з літературою про атаки сторонніми каналами на шифр AES та специфікаціями шифру	Вересень-жовтень 2023 р.	Виконано
3	Аналіз та огляд специфікацій шифру AES. Виконання оглядової частини дипломної роботи	Листопад-грудень 2023 р.	Виконано
4	Запропоновано нову модель атаки в результаті якої отримується ключ з помилковими бітами. Аналіз моделі, обрахунок ймовірності успішного виправлення помилок та складності виконання алгоритму	Січень-лютий 2024 р.	Виконано
5	Додаткове ознайомлення з літературою про атаки побічними каналами на шифр AES	Березень 2024 р.	Виконано
6	Програмна реалізація обрахунків ймовірності успішного виправлення помилок та автоматизація виводу отриманих результатів	Квітень2024 р.	Виконано
7	Апроксимація параметру q_m для великої кількості отриманих замірів ключа	Травень 2024 р.	Виконано

Студент _____ Толмачов Є.Ю.

Керівник _____ Савчук М.М.

РЕФЕРАТ

Кваліфікаційна робота містить: 76 стор., 2 рисунки, 11 таблиць, 11 джерел.

В даній роботі розглянуто криптографічний стандарт AES, поняття атак сторонніми каналами, доповнення атак на основі інформації зі сторонніх каналів на прикладі AES. Більшість таких атак засновано на статистичних методах та точних фізичних вимірах інформації з побічних каналів, через що отриманий в результаті атаки ключ може мати помилки та бути хибним. Метою роботи було дослідження алгоритмів виправлення помилок в знайденому в результаті атаки ключі. У ході роботи розглянуто три криптографічні моделі атаки на алгоритм AES за побічними каналами. Теоретично виведено та оцінено ймовірність успіху таких атак та їхню складність. Програмно реалізовано обрахунок оцінок ймовірності успіху та складності атак.

AES, КРИПТОАНАЛІЗ, АТАКА СТОРОННІМИ КАНАЛАМИ, ВИПРАВЛЕННЯ ПОМИЛОК

ABSTRACT

Qualification work contains: 76 pp., 2 figures, 11 tables, 11 sources.

In this paper cryptographic standard AES, the concept of side-channel attacks (SCA-attacks), and modifications of SCA-attacks with AES as an example were examined. Most of SCA-attacks are based on statistical methods and precise physical measurements of information from side channels. Due to this, the key obtained as a result of the attack may be distorted by errors. The aim of the work was to study the algorithms for correcting errors in the key that was found as a result of the attack. In the course of this work, three cryptographic models of SCA-attacks on the AES algorithm are examined. The probability of success of such attacks and their complexity were theoretically estimated and calculated. Software for calculating probability of success and complexity of said attacks is implemented.

AES, CRYPTOANALYSIS, SIDE-CHANNEL ATTACKS, ERROR CORRECTION

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SCA — Side-Channel Attack — атака на побічний канал

SPA — Simple Power Analysis — простий аналіз за енергоспоживанням

AES — Advanced Encryption Standard — блочний шифр AES

$\sum_{i=k}^{n}$ — сума від k до n

$\Pr\{A\}$ — ймовірність того що подія A відбудеться

$\oplus$ — операція побітового додавання

$\otimes$ — множення матриць

$\bullet$ — множення поліномів

$\lfloor x \rfloor$ — ціла частина параметра x

C_n^k — кількість комбінацій, кількість k-сполук з множини з n елементів

ВСТУП

Актуальність дослідження. Питання безпеки інформації є особливо важливим в сучасному цифровому світі і саме для вирішення цього питання використовуються криптографічні алгоритми. Нині наявні алгоритми є математично стійкими до багатьох видів атак, але існують джерела інформації про алгоритми що не залежать від їх математичного опису. Будь-який алгоритм виконується на фізичному пристрої - і в результаті роботи цього пристрою випромінюється величезна кількість додаткової інформації - тепло, час виконання, збої під час роботи тощо. Джерела такої інформації називають сторонніми каналами, і атаки на основі них зазвичай на порядок швидші від математичних атак.

Завдяки своїй надійності, швидкодії та малого використання ресурсів блокові шифри використовуються в багатьох сферах діяльності - в смарт-картах, у Wi-Fi роутерах, в інтернет-протоколах тощо. На момент 2024 року Advanced Encryption Standard є одним з найбільш вживаних криптографічних шифрів. Окрім того, AES є стійким до атак на основі квантових комп'ютерів. Саме тому є актуальним дослідження його стійкості до атак сторонніми каналами.

Зазвичай атаки сторонніми атаками є значно більш ефективними ніж математичні, але несуть можливість спотворення шуканого ключа через статистичні помилки. В даній роботі розглянуто криптографічний стандарт AES, поняття атаки сторонніми каналами, наведено деякі типи атак що існують для шифру AES та досліджено декілька алгоритмів виправлення помилок в хибному ключі на прикладі атаки на алгоритм AES.

1) провести огляд опублікованих джерел за тематикою дослідження;

2) сформулювати моделі та алгоритми;

3) теоретично вивести ймовірність успіху таких атак та їхню складність;

Об'єктом дослідження є системи криптографічного захисту інформації.

Предметом дослідження є алгоритми виправлення помилок в отриманому криптоаналітиком ключі.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: методи лінійної та абстрактної алгебри, теорії імовірностей, комбінаторного аналізу, теорії складності алгоритмів, методи комп'ютерного моделювання.

Практичне значення результатів полягає в можливості виправлення помилок хибного ключа та апроксимації часу та ймовірності успішного виправлення помилок.

1 АТАКИ НА СТОРОННІ КАНАЛИ ТА КРИПТОГРАФІЧНИЙ СТАНДАРТ AES

Цей розділ присвячено короткому огляду поняття SCA-атак та опису криптографічного стандарту AES.

1.1 SCA-атаки

В даній секції буде наведено деякі основні означення пов'язані з атаками на сторонні канали. Секція буде викладена відповідно до [2]

Означення 1.1. Атаки на побічний канал — клас атак що використовує вразливості фізичного пристрою на якому виконуються криптографічні операції.

Атаки бувають різними - за часом, за використаною потужністю, за шумом тощо. Будь-яка особливість фізичного пристрою та його середовища може дати інформацію про подальшу атаку.

Залежно від контролю за перебігом обчислювального процесу криптоаналітиком SCA-атаки можна розділити на наступні категорії:

– Пасивні атаки - атаки які непомітно втручаються у роботу цільової системи. Криптоаналітик отримує деяку інформацію про роботу цільової системи, але цільова система поводиться так само як і при відсутності атаки

– Активні атаки - атаки при яких криптоаналітик впливає на поведінку цільової системи. Навіть якщо система не може виявити різницю в роботі системи, то сторонній спостерігач може.

За типом доступу до пристрою атаки поділяються на

– Івазивні атаки - атаки що включають пряме втручання в внутрішні компоненти пристрою з можливим його пошкодженням. Це, наприклад, встановлення зчитувальної голки на шини даних. Деякі пристрої мають вбудовані протидії для подібних атак і змінюють свою

роботу в разі знаходження втручання.

– Напівінвазивні атаки передбачають наявність доступу до пристрою та можливістю втручання, але без його пошкодження. Це, наприклад, атака за збоями.

– Неінвазивні атаки - атаки що не передбачають втручань у пристрій. Це атаки засновані на спостереженні за роботою пристрою - наприклад, атаки за часом. Такі атаки неможливо визначити під час їх виконання.

Також деякі джерела виділяють віддалені атаки - це, наприклад, атаки за інтернет-трафіком, але в загальному випадку їм можна віднести до неінвазивних.

За типом методу аналізу атаки поділяються на прості атаки сторонніми каналами та диференційні.

1.2 огляд AES

Секція присвячена огляду шифру AES та викладена відповідно до [8]

Означення 1.2. AES (Advanced Encryption Standard) - ітераційний симетричний блоковий шифр з довжиною блоку 128 біт та ключем 128, 192 чи 256 біт.

Розмір вхідного блоку - N_b слів, кожне з яких - по 32 біти. Кількість раундів визначається параметром N_r. І розмір ключа - параметром N_k - кількістю 32-бітних слів. Комбінації параметрів що відповідають прийнятим стандартам зазначені в таблиці 1.1. Вхідний блок представлений у вигляді двовимірного масиву. Процес шифрування та розшифрування використовує 4 різні байтові операції: заміна байтів за допомогою таблиць заміни (S-box), зсув рядків стану блока, змішування даних з кожної колонки масиву стану, додавання раундового ключа.

Алгоритм 1.1. алгоритм шифрування AES

Вхід: $N_r, RoundKeys = rk_0, rk_1 ... rk_{N_r}$

1. Покласти $state = input$

Таблиця 1.1 – Комбінації параметрів шифру AES що відповідають стандарту

	Довжина ключа (N_k слів)	Довжина блоку (N_b слів)	Кількість раундів (N_r)
$AES-128$	4	4	10
$AES-192$	6	4	12
$AES-256$	8	4	14

2. $state = AddRoundKey(state, rk_0)$
3. Покласти $round = 1$
4. Поки $round < N_r$:
    5. $state = SubBytes(state)$
    6. $state = ShiftRows(state)$
    6. $state = MixColumns(state)$
    6. $state = AddRoundKey(state, rk_{round})$
5. $state = SubBytes(state)$
6. $state = ShiftRows(state)$
7. $state = AddRoundKey(state, rk_{N_r})$

Вихід: $state$

1.2.1 $AddRoundKey()$

Проста операція яка додає до поточного стану шифротексту байти ключів - до кожної колонки відповідний ключ

$$[s'_{0,c}, s'_{2,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{2,c}, s_{2,c}, s_{3,c}] \oplus rk_{round,c} \quad \text{для} \quad c = \overline{0, N_b}$$

1.2.2 $SubBytes()$

Операція нелінійної заміни байтів за допомогою таблиць заміни (S-коробки). Виконується на кожному байті не залежно від його позиції в масиві стану блоку.

$$b'_i = b_i \oplus b_{(i+4)mod8} \oplus b_{(i+5)mod8} \oplus b_{(i+6)mod8} \oplus b_{(i+7)mod8} \oplus c_i$$

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

1.2.3 $ShiftRows()$

Перетворення $ShiftRows()$ циклічно зсуває байти останніх трьох рядків стану на різне значення зсуву. Перший рядок без зсуву, оскільки значення зсуву $r = 0$.

Таким чином, можна записати як

$$s'_{r,c} = s_{r,(c+shift(r,N_b))modN_b} \quad \text{для} \quad 0 < r < 4, 0 \leqslant c \leqslant N_b$$

У випадку $N_b = 4$ значення $shift(r, 4) = r$.

1.2.4 $MixColumns()$

Перетворення $MixColumns()$ змінює стан шифротексту по колонках. Колонки вважаються поліномами над полем $GF(2^8)$ за модулем $x^4 + 1$ з фіксованим поліномом

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\},$$

де $\{03\}$ - байт в Шістнадцятковому представленні і відповідає двійковому запису поліному $\{00000011\} = x + 1$

Тоді перетворення можна представити у вигляді множення матриць $s'(x) = a(x) \otimes s(x)$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \qquad \text{для} \quad c = \overline{0, N_b}$$

В цьому перетворені байти колонки масиву замінюються наступним чином:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c})\ xor s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c})\ xor s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

1.2.5 Обернені функції

Аналогічно можна побудувати обернені функції для розшифрування тим самим ключем. $AddRoundKey()$ є сама собі оберненою. Для створення оберненої функції до $ShiftRows()$ необхідно

робити зсув в оберненому напрямі. Для оберненої функції до функції $SubBytes()$ необхідно використати обернену таблицю заміни байтів.

Для функції оберненої до $MixColumns()$ достатньо наступної операції:

$s'(x) = a^{-1}(x) \otimes s(x)$

де $a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{03\}$.

Що матрично записується як

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 01 & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \qquad \text{для} \quad c = \overline{0, N_b}$$

В цьому перетворені байти колонки масиву замінюються наступним чином:

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c})\ xor(\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$

$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c})\ xor(\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$

$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c})\ xor(\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c})\ xor(\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

Висновки до розділу 1

В цьому розділі було розглянуто поняття атак сторонніми каналами, а також симетричний блоковий шифр AES. Оскільки час роботи шифру не залежить від довжини повідомлення (довжина фіксована), складність виконання AES можна оцінити як $O(1)$. Шифр виконується на реальному фізичному пристрої, а тому випромінює інформацію за сторонніми каналами. Надалі розглянуто приклад деяких типів SCA-атак на шифр AES.

2 АТАКИ СТОРОННІМИ КАНАЛАМИ НА ШИФР AES

Цей розділ присвячено огляду різних SCA-атак на шифр *AES* за різними сторонніми каналами, а також деякі способи протидії ним.

2.1 Атака за кешуванням

В цій секції буде описано принцип атаки за кешуванням відповідно до [5]. В попередній роботі було розглянуто атаки за часом що працюють за принципом того що різні операції потребують різного часу виконання, що дозволяє провести статистичний аналіз замірів часу і побітово відновити ключ. Але навіть, на перший погляд, стійкі до атак за часом алгоритми шифрування можуть мати певні умови що створюють різницю в часі виконання на фізичному приладі. Такою умовою може бути наявність або відсутність кеш-пам'яті центрального процесора (надалі - просто кеш). Кеш - пам'ять - пам'ять процесора для прискорення роботи процесора з оперативною пам'яттю. Також там зберігаються копії нещодавно використаних інструкцій та даних для швидкого доступу без звертання до основної пам'яті.

Якщо під час шифрування наступна інструкція наявна в кешу то час доступу до неї центральним процесором буде меншим ніж при звертанні до оперативної пам'яті. Також, при отриманні даних та інструкцій з оперативної пам'яті (яких немає в кеш-пам'яті), вона дублюється в кеш. Оскільки пам'ять не є нескінченним ресурсом, то нові записи в кеш будуть видаляти старі. На основі подібних явищ можна побудувати атаки за доступом що за зміною кешу отримують нову інформацію та атаки за часом що заміряють різницю часу виконання різних операцій.

2.1.1 Атака за доступом

Атаки за доступом зазвичай виконуються наступним чином. Два процеси запускаються на одному процесорі. Перший процес - цільовий, він виконує криптографічну логіку (шифрування). Назвемо його обчислювач. Другий процес - шпигун, який намагається отримати інформацію про ключ обчислювача. Для отримання інформації процес-шпигун поступово видаляє кеш-пам'ять аби визначити які саме рядки кешу використовує обчислювач. Знаючи відповідний шифротекст шпигун зможе дізнатись інформацію про секретний ключ обчислювача. Наприклад, знаючи які рядки кешу не використовуються, зменшується кількість кандидатів на ключ для роботи з наявним в шпигуна шифротекстом.

2.1.2 Атака за часом

В атаках за часом процес-шпигун спостерігає за роботою обчислювача та отримує інформацію про ключ на основі апроксимації кількості запитів в кеш інформації якої там не було (що займає більше часу). Таким чином, наприклад, можна знайти найбільш ймовірного кандидата в байт ключа побудувавши аналіз залежності часу роботи шифру на певному шифротексті від байта ключа.

2.1.3 Стратегія атаки

Завдяки тому, що на останньому раунді не відбувається операції $MixColumns$, при наявності одного байту останнього раундового ключа k^{N_r} та байту шифротексту x^{N_r+1} можливо відтворити байт тексту з передостаннього раунду x_{N_r}, що дає можливість відтворити байт ключа від попереднього раунду k^{N_r-1}:

$$x_i^{N_r} = SubBytes^{-1}(ShiftRows^{-1}(x_j^{N_r+1} \oplus k_j^{N_r})).$$

Таким чином складність пошуку слова попереднього

Операція $MixColumns$ ускладнює відтворення тексту для наступних раундів.

$$x_i^{N_r-1} = SubBytes^{-1}(ShiftRows^{-1}(MixColumns^{-1}(y_i))), \tag{2.1}$$

$$\text{де} \quad y_i = \{x_j^{N_r}|x_{j+1}^{N_r}|x_{j+2}^{N_r}|x_{j+2}^{N_r}\} \oplus \{k_j^{N_r-1}|k_{j+1}^{N_r-1}|k_{j+2}^{N_r-1}|k_{j+2}^{N_r-1}\}$$

Складність перебору стає 2^{32} - адже необхідно підібрати щонайменше 4 байти. Така складність для пошуку одного попереднього байта не є практичною якщо взяти до уваги збір статистичних даних для аналізу залежності часу роботи шифру від байта ключа - навіть 10000 замірів для 4 байтів ключа призведе до складності $O(2^{13+32})$. Одним з способів зменшення перебору є використання факту того що $MixColumns$ - лінійна операція, а отже має властивості гомоморфізму.

$$MixColumns^{-1}(X \oplus K) = MixColumns^{-1}(X) \oplus MixColumns^{-1}(K)$$

Тоді можна представити вираз як

$$x_i^{N_r-1} = SubBytes^{-1}(ShiftRows^{-1}(x_j'^{N_r} \oplus k_j'^{N_r-1})),$$

де $x_j'^{N_r}$— байт з $MixColumns^{-1}(\{x_j^{N_r}|x_{j+1}^{N_r}|x_{j+2}^{N_r}|x_{j+2}^{N_r}\})$, для пошуку $k_j'^{N_r-1} = MixColumns^{-1}(\{k_j^{N_r-1}|k_{j+1}^{N_r-1}|k_{j+2}^{N_r-1}|k_{j+2}^{N_r-1}\})$. Тоді, знайшовши всі $k_j'^{N_r-1}$, можливо відтворити весь раундовий ключ K_{N_r-1}. Складність пошуку для кожного байту буде 2^8, значить, складність перебору можливо знизити $O(2^8)$.

Варто зазначити що подібна статистична атака можлива не лише для атак за кешуванням, але й для атак по аналізу енергоспоживання, електромагнітним хвилям тощо. Кількість статистичних замірів необхідних для успішної атаки залежить від пристрою, типу стороннього

каналу та особливостей реалізації шифру AES.

2.2 Атака по аналізу енергоспоживання

В цій секції описано атаку по аналізу енергоспоживання на шифр AES відповідно до [3]. Найпростішою атакою за енергоспоживанням є SPA (Simple Power Analysis) атака - інтерпретація даних отриманих під час спостережень за енергоспоживанням під час криптографічних операцій. З них, наприклад, можливо ви. Кожен момент заміру енергоспоживання за певний момент часу називають слідом. Зазвичай заміри зашумлені, але, на щастя, шум зазвичай піддається нормальному розподілу.

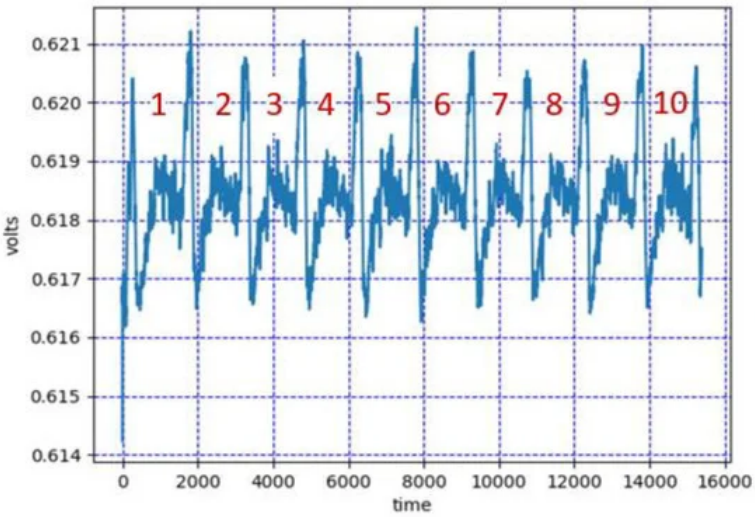

Рисунок 2.1 – SPA сліди показують пораундову періодичність енергоспоживання шифру AES

За графіком видно періодичність енергоспоживання. За фрагментами можливо встановити яка саме операція проводилась під час заміру. За часом (кількістю замірів) можливо знайти в яких випадках відбувається адресація до оперативної пам'яті та який з байтів ключа більш статистично ймовірний при наявних замірах.

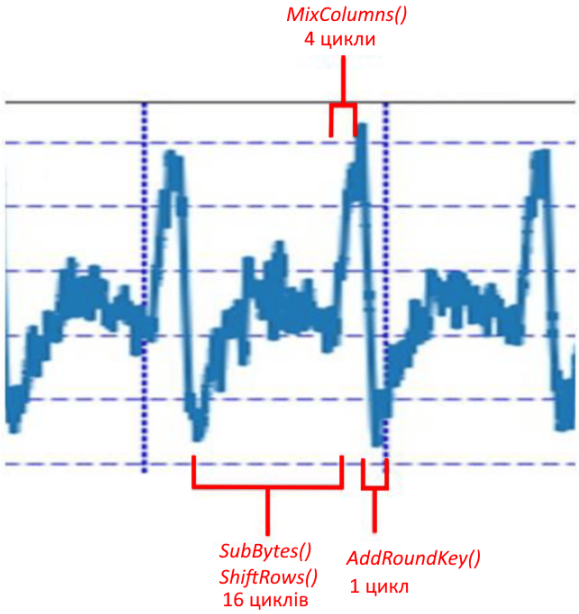

Рисунок 2.2 – SPA сліди дозволяють апроксимувати місця та час виконання функцій

2.2.1 Шаблонна атака

Шаблонна атака - атака заснована на порівнянні отриманих з пристрою вимірів з прикладами того як цей пристрій має працювати і на основі цього отримує інформації про ключ можливий. Для атаки в аналітика має бути доступ до копії цільового пристрою. Перед атакою створюється шаблон - тисячі слідів збираються для побудови відповідних статистичних моделей.

Атака відбувається в 4 етапи:

1. За допомогою копії пристрою шляхом використання комбінацій випадкових відкритих текстів та ключів зібрати величезну вибірку слідів енергоспоживання. Вибірка має бути достатньою для розрізнення кожного слова ключа.

2. Побудувати шаблон роботи пристрою. Шаблон має виділяти більш впливові заміри слідів та створювати розподіл цих слідів залежно від ключа.

3. Зібрати малу кількість слідів з цільового пристрою.

4. Застосувати шаблон до отриманих слідів. Необхідно обрати найбільш ймовірний підключ для кожного ключа.

В найпростішому варіанті шаблон є таблицею в якій кожному ключу відповідає розподіл, й на основі розподілів обирається найбільш ймовірний ключ.

З недоліків варто зазначити необхідність величезної кількості розподілів та слідів для статистичного аналізу. Необхідно мати десятки тисяч замірів для статистичного аналізу одного байту ключа. Цей недолік можна полегшити шляхом зміни цілі зі знаходження власне байту ключа до знаходження його ваги (зменшує кількість необхідних для побудови розподілів з 256 до 9).

2.3 Атака по випадковим збоям

Наступна секція буде викладена відповідно до [4]. Девайси які ми використовуємо вважаються надійними, але помилки заліза все ж існують і створюють певні ризики безпеки.

Існує багато способів створити помилки на приладі - створення надто високої напруги тощо. Надалі розглянемо способи створення помилок для подальших використань.

Зазвичай смарт-карти розраховані на контакт з поверхнею з напругою 4-5 вольт і можуть витримати до 10% додаткової напруги без проблем. Тому, можливо побудувати атаку на основі перевантаження. Така атака залежить від часу, напруги та форми транзистора напруги. Атака не є інвазивною і не потребує ніякої модифікації смарт-карти, але є напівінвазивною і її можливо виявити. Подібною атакою також є атака збою (glitch attack), котра побудована на зміні вольтажу - відхилення часу тактового сигналу може призвести до зовсім іншої роботи обчислювального пристрою.

Також можливо створення помилок шляхом перевертання бітів

оптичними засобами (спалахи, лазери тощо), але це є більш складним і випадковим процесом через складність підбору конкретного місця та часу для перевертання біта аби отримати необхідну інформацію.

Існує ще атака за електромагнітними спотвореннями. За допомогою проведення струму у котушці поруч з провідною поверхнею можливо перевернути біти в конкретний момент часу. Тоді помилки можна створювати помилки в необхідний криптоаналітику час. Також напівінвазивна атака.

2.4 Протидія атакам по стороннім джерелам

Дана секція викладена переважно за [7, 6, 2].

Існує багато способів протидії подібним атакам, але більшість з них лише збільшує кількість необхідних статистичних вимірів.

Основними та найпростішими способами протидії атакам сторонніми каналами - "засліплення"та зменшення "витоку"інформації сторонніми каналами з фізичних носіїв.

Додаючи випадкові операції можливо збільшити час чи енергоспоживання операцій, але зазвичай це "засліплення"формує певні розподіли та все ще піддається статистичному чи диференційному аналізу. Недоліком такого підходу є необхідність додаткових операцій, що збільшує складність виконання шифру, а також може створити нові непередбачувані вразливості.

Інший спосіб - зменшення розміру сигналів за допомогою зміни будови мікропроцесора (інші матеріали, їхня товщина) та впровадження константного часу виконання операцій. Це потребує додаткових ресурсів для розробки та виробництва таких мікропроцесорів що не завжди є оптимальним, та, оскільки неможливо повністю нівелювати витік інформації, лише збільшить кількість необхідних статистичних вимірів.

Одним зі способів протидії також є "балансування"для маскування одного набору дій іншим "доповнюючим". Наприклад, якщо всі байти

мають вагу 4, то на кожну операцію з бітом 1 прийде операція з бітом 0. Але, оскільки момент виконання операцій не є однаковим, то збільшивши кількість статистичних вимірів можливо побудувати модель що бере до уваги наявність "доповнюючого"набору дій.

Висновки до розділу 2

В цьому розділі було розглянуто ряд атак сторонніми каналами на блоковий шифр. Зазначені атаки мають схожий принцип, а саме - знаходження моментів під час яких відбувається звертання до оперативної пам'яті та статистичний збір даних для побудови моделей за якими можливо підібрати найбільш ймовірні байти шуканого ключа. Також було розглянуто способи протидії атакам. Більшість методів протидії атакам сторонніми атаками потребують використання значної кількості додаткових ресурсів (або під час виконання або під час виготовлення обчислюючих пристроїв) при тому не гарантуючи значного ускладнення виконання атак.

3 АТАКА З ВИПРАВЛЕННЯМ ПОМИЛОК

В цьому розділі буде розглянуто декілька моделей атаки що відбулась та алгоритмів виправлення помилок в знайденому криптоаналітиком ключі з помилками $\tilde{r}$. Нехай відбулася SCA-атака на AES. Перші дві моделі є модифікаціями моделей для криптографічного шифру RSA з [11].

3.1 Модель 1

Нехай модель 1 визначається наступним чином:

1. Відбулась атака за побічними каналами на основі шифротексту

2. Криптоаналітик E, маючи доступ до пристрою що використовує шифр AES та маючи інформацію з побічного каналу про виконання операцій

$$C = AES(M, r), \tag{3.1}$$

$$\text{або} \quad M = InvAES(C, r), \tag{3.2}$$

де $AES(M, r)$ — операція шифрування повідомлення M ключем r, та $InvAES(C, r)$ — операція розшифрування шифротексту C ключем r, отримав біти секретного ключа з розміром β бітів та з деякою помилкою

$$\tilde{r} = \tilde{r}_{\beta-1}, \tilde{r}_{\beta-2}, ..., \tilde{r}_2, \tilde{r}_1, \tilde{r}_0, \quad \tilde{r}_i \in \{0,1\}. \tag{3.3}$$

3. Вважаємо, що

$$\tilde{r}_i = r_i \oplus \varepsilon_i,$$

де $r_i \in \{0, 1\}$ — істинне значення біту, $\varepsilon_i \in \{0, 1\}$ — випадкові незалежні величини. Нехай ймовірність помилки в кожному біті визначається таким чином:

$$\Pr\{\varepsilon_i = 1\} = p, \quad 0 \leqslant p \leqslant \frac{1}{2}, \quad i = \overline{0, \beta - 1}.$$

Якщо $p = 0$, то помилок немає і $\tilde{r}$ — істинний ключ r. Якщо ж $p = \frac{1}{2}$, то з (3.3) знайти якусь інформацію про істинний ключ r неможливо.

4. Криптоаналітик E може перевірити чи є $\tilde{r}$ істинним ключем чи ключем з помилками підстановкою в рівняння (3.1) або (3.2).

3.1.1 Алгоритм атаки

Мета другої половини атаки — із ключа з помилками (3.3) знайти справжній ключ r. При цьому в (3.3) можливі $1, 2, ..., k, ..., \beta$ помилок.

Алгоритм 3.1. Алгоритм виправлення до k помилок

Вхід: $\tilde{r}$, M, C, k

Вихід: r

1. Якщо $AES(M, \tilde{r}) = C$, то

    2. $r = \tilde{r}$, алгоритм завершує роботу.

3. $i = 1$

4. Поки $i \leqslant k$

    5. Визначити Cm_i

    6. Для всіх j з Cm_i

        7.$\tilde{r'} = \tilde{r} \oplus j$

        8. Якщо $AES(M, \tilde{r}) = C$, то

            9. $r = \tilde{r'}$, алгоритм завершує роботу.

    10. $i = i + 1$

11. Повернути помилку, алгоритм завершує роботу.

Тут Cm_i — множина усіх бітових рядків довжини $|\tilde{r}|$ з вагою i. Множину Cm_i можна представити у вигляді послідовності всіх можливих комбінацій i елементів з β у лексикографічному порядку. З деякими алгоритмами побудови лексикографічної послідовності можна ознайомитись в [9].

В якості рівняння для перевірки істинності ключа окрім (3.1) можна використовувати рівняння (3.2).

Помилок в ключі може бути більше k, отже алгоритм має певну ймовірність успіху.

3.1.2 Аналіз алгоритму

Теорема 3.1. *Ймовірність успіху алгоритму 3.1 що виправляє до k помилок буде дорівнювати*

$$\Pr\{\textit{успіху}\} = \sum_{i=0}^{k} C_{\beta}^{i} \cdot p^{i} \cdot (1-p)^{\beta-i}$$

Доведення.

$$\Pr\{\text{Успіху}\} = \Pr\{x \leqslant k\},$$

де x — кількість помилок в $\tilde{r}$.

$$\Pr\{x \leqslant k\} = \sum_{i=0}^{k} \Pr\{x = i\} = \sum_{i=0}^{k} C_{\beta}^{i} \cdot p^{i} \cdot (1-p)^{\beta-i}$$

□

Ймовірність успіху при різних значеннях (β, p, k) наведено у таблиці 3.1.

Теорема 3.2. *Складність T_k виконання алгоритму 3.1 в операціях виконання шифру AES в найгіршому випадку дорівнює*

$$T_k = \sum_{i=0}^{k} C_{\beta}^{k}.$$

Доведення. Кількість виконань шифру AES необхідних для виправлення рівно i помилок — C_{β}^{i}. Тоді кількість операцій необхідних для виправлення до k помилок - $\sum_{i=0}^{k} C_{\beta}^{k}$ що і є складністю в найгіршому випадку. □

Складність виконання алгоритму в найгіршому випадку при різних значеннях (p, β, k) наведено у таблиці 3.2.

Таблиця 3.1 – Ймовірність успіху атаки в моделі 1 при ймовірності помилки p, розмірі ключа β та k виправлень

$\beta = 128$				
k =	1	2	5	10
$p = 0.01$	0.6334	0.8625	0.9981	1.0
$p = 0.1$	0.0	0.0002	0.0093	0.256
$p = 0.15$	$2^{-25.45}$	$2^{-21.9}$	0.0001	0.011
$p = 0.3$	$2^{-60.06}$	$2^{-55.27}$	$2^{-43.86}$	$2^{-30.09}$
$\beta = 192$				
k =	1	2	5	10
$p = 0.01$	0.4268	0.6984	0.9866	1.0
$p = 0.1$	$2^{-24.7}$	$2^{-21.23}$	0.0001	0.0127
$p = 0.15$	$2^{-39.89}$	$2^{-35.77}$	$2^{-26.36}$	0.0
$p = 0.3$	$2^{-92.42}$	$2^{-87.05}$	$2^{-73.88}$	$2^{-57.11}$
$\beta = 256$				
k =	1	2	5	10
$p = 0.01$	0.2737	0.5278	0.9547	0.9999
$p = 0.1$	$2^{-34.03}$	$2^{-30.16}$	$2^{-21.45}$	0.0002
$p = 0.15$	$2^{-54.49}$	$2^{-49.97}$	$2^{-39.33}$	$2^{-26.73}$
$p = 0.3$	$2^{-124.94}$	$2^{-119.15}$	$2^{-104.74}$	$2^{-85.86}$

Таблиця 3.2 – Складність атаки в найгіршому випадку для моделей 1, 2, 3, в операціях виконання шифру AES

$\beta = 128$				
k =	1	2	5	10
C_β^k	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$\beta = 192$				
k =	1	2	5	10
C_β^k	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$\beta = 256$				
k =	1	2	5	10
C_β^k	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$

Теорема 3.3. *Складність виконання алгоритму 3.1 в середньому випадку*

$$\overline{T}_k = p_0 + \sum_{i=1}^{k} p_i \cdot \left(\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i \right) + \sum_{i=0}^{k} C_\beta^i \left(1 - \sum_{i=0}^{k} p_i \right),$$

де $p_i = C_\beta^i \cdot p^i \cdot (1-p)^{\beta - i}$.

Доведення. Складність в середньому буде дорівнювати сумі складності при певному випадку помножити на ймовірність цього випадку.

Ймовірність того що помилок буде $i \leqslant k$:

$$p_i = C_\beta^i \cdot p^i \cdot (1-p)^{\beta - i}$$

Ймовірність того що помилок більше k:

$$\left(1 - \sum_{i=0}^{k} p_i \right)$$

Кількість операцій при кількості помилок $i = 0 : 1$ виконання шифру AES.

Кількість операцій при кількості помилок $i \leqslant k$: $\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_{\beta+1}^i$ виконань шифру AES.

Кількість операцій при кількості помилок $i > k$: $\sum_{i=0}^{k} C_\beta^i$ виконань шифру AES. Тоді загальна складність в середньому:

$$p_0 + \sum_{i=1}^{k} p_i \cdot \left(\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i \right) + \sum_{i=0}^{k} C_\beta^i \left(1 - \sum_{i=0}^{k} p_i \right)$$

□

Складність виконання алгоритму в середньому випадку при різних значеннях (p, β, k) наведено у таблиці 3.3.

Дана модель припускає наявність лише одного джерела та одного

заміру ключа. На практиці зазвичай буде можливість отримати більшу кількість замірів ключа, що може значуще покращити ймовірність успіху атаки та потенційно зменшити складність виправлення помилок.

Таблиця 3.3 – Складність атаки для моделі 1 в середньому з довжиною ключа β та k виправлень

$\beta = 128$				
k =	1	2	5	10
$p = 0.01$	$2^{6.15}$	$2^{11.05}$	$2^{20.8}$	$2^{27.8}$
$p = 0.1$	$2^{7.01}$	$2^{13.01}$	$2^{28.03}$	$2^{47.49}$
$p = 0.15$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.8}$
$p = 0.3$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$\beta = 192$				
k =	1	2	5	10
$p = 0.01$	$2^{7.11}$	$2^{12.99}$	$2^{25.95}$	$2^{38.31}$
$p = 0.1$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.78}$
$p = 0.15$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$p = 0.3$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$\beta = 256$				
k =	1	2	5	10
$p = 0.01$	$2^{7.73}$	$2^{14.27}$	$2^{29.47}$	$2^{45.8}$
$p = 0.1$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$p = 0.15$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$p = 0.3$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$

Наступні моделі розглядають випадок наявності більше одного заміру ключа.

3.2 Модель 2

Припущення моделі 2

Розглянемо другу модель. Нехай криптоаналітик E знає не одне, а декілька рівнянь типу (3.1) та (3.2).

$$C_j = AES(M_j, r), \tag{3.4}$$

$$M_j = InvAES(C_j, r), \tag{3.5}$$

І відповідно

$$\tilde{r^j} = \tilde{r^j}_{\beta-1}, \tilde{r^j}_{\beta-2}, ..., \tilde{r^j}_2, \tilde{r^j}_1, \tilde{r^j}_0,$$
$$\tilde{r^j}_i \in \{0,1\}, \quad j \in \{1, m\},$$
$$\tilde{r_i^j} = r_i^j \oplus \varepsilon_i^j,$$

де $r_i^j \in \{0,1\}$ — істинне значення біту, $\varepsilon_i^{(j)} \in \{0,1\}$ — випадкові незалежні величини. $Pr\{\varepsilon_i^{(j)} = 1\} = p$, $0 \leqslant p \leqslant \frac{1}{2}$, $i = \overline{0, \beta}, \quad j = \overline{1, m}$.

Алгоритм 3.2. Алгоритм виправлення до k помилок для моделі 2.

Вхід: M, C, k, $\tilde{r}$, $j = \overline{1, m}$

Вихід: r

1. $i = 0$

2. Поки $i \leqslant \beta$

3. $\tilde{r}_i = \begin{cases} 1, & \sum\limits_{j=1}^{m} \tilde{r}_i^{(j)} \geqslant \dfrac{m}{2} \\ 0, & \sum\limits_{j=1}^{m} \tilde{r}_i^{(j)} < \dfrac{m}{2} \end{cases}$

4. $i = i + 1$

5. Запустити алгоритм 3.1 з параметрами $\tilde{r}$, M, C, та k

Навіть після таких маніпуляцій результуючий ключ може залишитись з помилками, а отже, виконання алгоритму може не призвести до бажаного результату.

3.2.1 Аналіз алгоритму

Теорема 3.4. *Ймовірність успіху алгоритму 1.1 що виправляє до k помилок буде дорівнювати*

$$\Pr\{\textit{Успіху}\} = \sum_{i=0}^{k} C_{\beta}^{i} \cdot q_m^i \cdot (1-q_m)^{\beta-i},$$

$$\textit{де} \qquad q_m = \sum_{i=0}^{\lfloor m/2 \rfloor} C_m^i \cdot p^{m-i} \cdot (1-p)^i$$

Доведення.

$$\Pr\{\text{Успіху}\} = \Pr\{x \leqslant k\},$$

де x — кількість помилок в $\tilde{d}$.

$$\Pr\{x \leqslant k\} = \sum_{i=0}^{k} \Pr\{x = i\}$$

Оскільки наявно m рівнянь, хоча б $\frac{m}{2}$ з них мають мати помилку у i-тому біті для того аби була помилка в біті $\tilde{r_i}$, тому ймовірність того що $\tilde{r_i} = r_i \oplus 1$ дорівнює ймовірності того що помилка в біті наявна в половині ключів: $\tilde{r_i}^{(j)} = r_i^{(j)} \oplus 1$. Нехай ймовірність того що помилка наявна в половині ключів дорівнює q_m, тоді

$$q_m = \Pr\{y \geqslant \frac{m}{2}\} = \sum_{i=0}^{\lfloor m/2 \rfloor} C_m^i \cdot p^{m-i} \cdot (1-p)^i,$$

де y — кількість помилок в j-тих бітах ключів $\tilde{r_j}^{(1)}$,...,$\tilde{r_j}^{(r)}$

$$\Pr\{y = i\} = C_m^i \cdot p^{m-i} \cdot (1-p)^i$$

$$\Pr\{\text{Успіху}\} = \Pr\{x \leqslant k\} = \sum_{i=0}^{k} C_{\beta}^{i} \cdot q_m^i \cdot (1-q_m)^{\beta-i}$$

□

Ймовірність успіху при різних значеннях (β, p, k, m) буде наведено у таблиці Б.1.

Теорема 3.5. *Складність виконання алгоритму 1.1 в найгіршому випадку*

$$\overline{T}_k = m \cdot \beta \quad \textit{додавань} + \sum_{i=0}^{k} C_\beta^i \quad \textit{виконань шифру AES}$$

Доведення. Кількість необхідних операцій додавань бітів — m додавань на кожен з β бітів.

Складність додавання — $O(1)$, тому складність додавання $O(m \cdot \beta)$

Доведення для кількості виконань операцій в найгіршому випадку аналогічно доведенню для моделі 1. □

Кількістю додавань можна знехтувати. Складність в найгіршому випадку наведено в таблиці 3.2.

Теорема 3.6. *Складність виконання алгоритму 3.2 в середньому*

$$\overline{T}_k = x_0 + \sum_{i=1}^{k} x_i \cdot \left(\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i \right) + \sum_{i=0}^{k} C_\beta^i \left(1 - \sum_{i=0}^{k} x_i \right) \tag{3.6}$$

де $x_i = C_\beta^i \cdot q_m^i \cdot (1 - q_m)^{\beta - i}, \quad q_m = \sum_{i=0}^{\lfloor m/2 \rfloor} C_m^i \cdot p^{m-i} \cdot (1-p)^i$

Доведення. Додавання на початку алгоритму робляться у будь-якому випадку.

Доведення складності множень аналогічно доведенню теореми 3.1, із розподілом $Bin(p = q_m, \beta)$. □

Складність в середньому випадку для різних параметрів (β, p, k, m) наведено в таблиці Б.2.

3.3 Модель 3

Припущення моделі 3

Нехай криптоаналітик E знає не одне, а декілька рівнянь типу (3.1) та (3.2) з різних джерел.

$$C_j = AES(M_j, r), \tag{3.7}$$

$$M_j = InvAES(C_j, r), \tag{3.8}$$

І відповідно

$$\tilde{r^j} = \tilde{r^j}_{\beta-1}, \tilde{r^j}_{\beta-2}, ..., \tilde{r^j}_2, \tilde{r^j}_1, \tilde{r^j}_0,$$
$$\tilde{r^j}_i \in \{0,1\}, \quad j \in \{1, m\},$$
$$\tilde{r_i^j} = r_i^j \oplus \varepsilon_i^j,$$

де $r_i^j \in \{0,1\}$ — істинне значення біту, $\varepsilon_i^j \in \{0,1\}$ — випадкові незалежні величини,

$$\Pr\{\varepsilon_i = 1\} = p_j, \quad 0 \leqslant p_j \leqslant \frac{1}{2}, \quad i = \overline{0, \beta - 1}.$$

Алгоритм 3.3. Алгоритм виправлення до k помилок для моделі 2.

Вхід: M, C, k, $\tilde{r}$, $j = \overline{1, m}$

Вихід: r

1. $i = 0$

2. Поки $i \leqslant \beta$

3. $\tilde{r}_i = \begin{cases} 1, & \sum\limits_{j=1}^{m} \tilde{r}_i^{(j)} \geqslant \dfrac{m}{2} \\ 0, & \sum\limits_{j=1}^{m} \tilde{r}_i^{(j)} < \dfrac{m}{2} \end{cases}$

4. $i = i + 1$

5. Запустити алгоритм 3.1 з параметрами $\tilde{r}$, M, C, та k

Ця модель також може мати більше k помилок, тому атака на неї має певну ймовірність успіху.

3.3.1 Аналіз алгоритму

Ймовірність успіху алгоритму 3.3 що виправляє до k помилок дорівнює

$$\Pr\{\text{успіху}\} = \sum_{i=0}^{k} C_{\beta}^{i} \cdot q_m^i \cdot (1 - q_m)^{k-i},$$

де

$$q_m = \sum_{i_1+i_2+\ldots+i_m \leqslant \lfloor m/2 \rfloor} p_1^{1-i_1} \ldots p_n^{1-i_m}.$$

Доведення.

$$\Pr\{\text{Успіху}\} = \Pr\{x \leqslant k\},$$

де x — кількість помилок в $\tilde{d}$.

$$\Pr\{x \leqslant k\} = \sum_{i=0}^{k} \Pr\{x = i\}$$

Оскільки в нас є r рівнянь, хоча б $\frac{m}{2}$ з них мають мати помилку у i-тому біті для того аби була помилка в біті $\tilde{r}_i$, тому ймовірність того що $\tilde{r}_i = r_i \oplus 1$ дорівнює ймовірності того що помилка в біті наявна в половині ключів: $\tilde{r}_i^{(j)} = r_i^{(j)} \oplus 1$. Нехай ймовірність того що помилка наявна в половині ключів дорівнює q_m, тоді

$$q_m = \Pr\{y \geqslant \frac{m}{2}\} = \sum_{i_1+i_2+\ldots+i_m \leqslant \lfloor m/2 \rfloor} p_1^{1-i_1} \ldots p_n^{1-i_m},$$

де y — кількість помилок в j-тих бітах ключів $\tilde{r}_j^{(1)}$,...,$\tilde{r}_j^{(r)}$

$$\Pr\{y = j\} = p_1^{1-i_1} \ldots p_m^{1-i_m}$$

$$\Pr\{\text{Успіху}\} = \Pr\{x \leqslant k\} = \sum_{i=0}^{k} C_\beta^i \cdot q_m^i \cdot (1-q_m)^{k-i}$$

□

Ймовірність успіху при різних значеннях (β, p, m, k) наведено у таблиці Б.3.

Теорема 3.7. *Складність алгоритму що виправляє до k помилок в найгіршому випадку буде дорівнювати*

$$T_k = \sum_{i=0}^{k} C_\beta^k$$

Доведення аналогічне до доведення для алгоритму 3.1. Складністю додавань було знехтувано. Складність наведено в таблиці 3.3

Теорема 3.8. *Складність виконання алгоритму виправлення до k помилок в середньому*

$$\overline{T}_k = x_0 + \sum_{i=1}^{k} x_i \cdot \left(\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i \right) + \sum_{i=0}^{k} C_\beta^i \left(1 - \sum_{i=0}^{k} x_i \right).$$

де $x_i = C_\beta^i \cdot q_m^i \cdot (1-q_m)^{\beta-i}, q_m = \Pr\{y \geqslant \frac{m}{2}\} = \sum\limits_{i_1+i_2+\ldots+i_m \leqslant \lfloor m/2 \rfloor} p_1^{1-i_1} \ldots p_m^{1-i_m}.$

Доведення.

Складність в середньому буде дорівнювати сумі складності при певному випадку помножити на ймовірність цього випадку.

Ймовірність того що помилок буде $i \leqslant k$:

$$x_i = \sum_{i=0}^{k} C_\beta^i \cdot q_m^i \cdot (1-q_m)^{\beta-i}$$

$$\text{де} \quad q_m = \Pr\{y \geqslant \frac{m}{2}\} = \sum_{i_1+i_2+\ldots+i_m \leqslant \lfloor m/2 \rfloor} p_1^{1-i_1} \ldots p_m^{1-i_m},$$

Ймовірність того що помилок більше k:

$$\left(1 - \sum_{i=0}^{k} x_i\right)$$

Кількість операцій при кількості помилок $i = 0 : 1$ виконання шифру AES.

Кількість операцій при кількості помилок $i \leqslant k$: $\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i$ виконань шифру AES.

Кількість операцій при кількості помилок $i > k$: $\sum_{i=0}^{k} C_\beta^i$ виконань шифру AES.

Тоді загальна складність

$$x_0 + \sum_{i=1}^{k} x_i \cdot \left(\sum_{j=0}^{i-1} C_\beta^j + \frac{1}{2} C_\beta^i\right) + \sum_{i=0}^{k} C_\beta^i \left(1 - \sum_{i=0}^{k} x_i\right) \text{ виконань шифру } AES.$$

□

Також алгоритм 3.3 потребує $m \cdot \beta$ додавань. Складністю додавань можна знехтувати. В такому разі в найгіршому випадку складність в операціях множеннях дорівнює складності алгоритму 3.1. Але ймовірність успіху буде значно більшою. Варто зазначити що через особливість 3 пункту 3.3, ймовірність успіху при наявності непарної кількості $2k - 1$ або $2k + 1$ рівнянь буде значно вищою ніж при використанні $2k$ рівнянь.

Варто зазначити що модель підходить для будь-якого шифру. Оцінка складності та ймовірності успіху також. При подальшому виконанні роботи буде розглянуто різні моделі задання ймовірностей з кожного джерела (експоненційне зростання ймовірності, лінійне тощо). Програмна реалізація підрахунку ймовірності успіху алгоритму 3.3 та його складності буде наведено в 3.4.2.

3.4 Гранична теорема для оцінки алгоритмів виправлення до к помилок моделей 2 та 3

3.4.1 Теоретичні викладки

Нехай в нас є значно більша кількість рівнянь вигляду 3.7 або 3.8. Тоді оцінити ймовірність успіху алгоритмів виправлення помилок та їхню середню складність стає значно складніше, особливо для моделі 3. Основною проблемою є параметр q_m. Тому можна апроксимувати ймовірність успіху. Дана підсекція буде викладена відповідно до [10].

Дано незалежну схему випробувань $A_0, A_2, ...A_{m-1}$, з $P(A_k) = p_k, k = \overline{0,m-1}$, S_n - число успіхів випробувань. Якщо при $n \longrightarrow \infty$ сума $\sum\limits_{k=0}^{n} p_k(1-p_k) \longrightarrow \infty$, то

$$\Pr\{\frac{S_n - \sum_{k=0}^{n} p_k}{\sqrt{\sum_{k=0}^{n} p_k(1-p_k)}} < x\} \longrightarrow \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-y^2}{2}} dy,$$

$$\text{де} \quad MS_n = \sum_{k=0}^{n} p_k, \quad DS_n = \sum_{k=0}^{n} p_k(1-p_k)$$

Нехай при $n \longrightarrow \infty$ для будь-якого k : $\quad p_k \geqslant 0$ і $\lambda = \sum_{i=0}^{m-1} p_i$
Тоді $\Pr\{S_n = h\} \longrightarrow \dfrac{\lambda_m^h e^{-\lambda_m}}{h!}$

3.4.2 Апроксимація параметру q_m

Лема 3.1. *Для моделі 2 параметр λ_m має вигляд*

$$\lambda_m = m \cdot p$$

Лема 3.2. *Для моделі 3 параметр λ_m має вигляд*

$$\lambda_m = \sum_{i=0}^{m-1} p_i$$

Теорема 3.9. *Параметр q_m для моделей 2 та 3 при великій кількості рівнянь можна прирівняти до*

$$q_m = 1 - \sum_{h=0}^{\lfloor m/2 \rfloor} \frac{\lambda_m^h e^{-\lambda_m}}{h!}$$

Доведення. q_m — ймовірність спотворення біту після виконання пункту 3 з алгоритму 3.2. Тоді

$$q_m = \Pr\{S_n > \lfloor \frac{m}{2} \rfloor\} = 1 - \Pr\{S_n \leqslant \lfloor \frac{m}{2} \rfloor\}$$

З 3.4.1 $\Pr\{S_n = h\} \longrightarrow \dfrac{\lambda_m^h e^{-\lambda_m}}{h!}$, тоді

$$q_m = \sum_{h=\lceil m/2 \rceil}^{m} \frac{\lambda_m^h e^{-\lambda_m}}{h!} = 1 - \sum_{h=0}^{\lfloor m/2 \rfloor} \frac{\lambda_m^h e^{-\lambda_m}}{h!}$$

$$q_m = 1 - \sum_{h=0}^{\lfloor m/2 \rfloor} \frac{\lambda_m^h e^{-\lambda_m}}{h!}$$

□

Таблиця 3.4 – Параметр q_m для різних p, m

$m =$	25	51	101	201
$p = 0.3$	0.0427	0.0078	0.0004	$2^{-19.83}$
$m =$	25	51	101	201
$p = 0.4$	0.2084	0.1309	0.0601	0.0148
$m =$	25	51	101	201
$p = 0.45$	0.3389	0.2887	0.2236	0.1457

Апроксимація параметру q_m, ймовірність успіху та складність в середньому випадку алгоритмів 3.2, 3.3 буде наведено в таблицях 3.4, Б.5, Б.6.

Висновки до розділу 3

Цей розділ був присвячений опису та аналізу алгоритмів виправлення помилок для трьох моделей. Було запропоновано та проаналізовано алгоритм для трьох моделей - з одним виміром хибного ключа, з більшою кількістю вимірів ключа та з виміром хибного ключа з різних джерел. З отриманих результатів можна зробити висновок - наявність більше ніж одного ключа значно покращує ймовірність успіху алгоритму. Варто зазначити що, через особливості алгоритму, непарна кількість замірів для другої та третьої моделі $(m = 2n - 1, m = 2n + 1)$ дає більшу ймовірність успіху для алгоритму ніж парна $(m = 2n)$. Також запропоновано спосіб апроксимації параметра для оцінки алгоритму в випадку з великою кількістю замірів ключа.

ВИСНОВКИ

В ході роботи було надано короткий огляд основних понять пов'язаних з SCA-атаками, наведено специфікації шифру AES та приклади можливих атак сторонніми каналами на нього. З масовим використанням смарт-карток та мережі інтернет питання атак сторонніми каналами на цей шифр набувають значної актуальності.

Також було запропоновано три моделі атаки що відбулася для подальшого виправлення помилок - модель з одним заміром хибного ключа, модель з декількома замірами хибного ключа та модель з атакою за різними джерелами. Було проаналізовано алгоритми виправлення помилок хибного ключа отриманого в кожній моделі атаки що відбулася. Наведено спосіб наближеної оцінки складності та ймовірності успіху алгоритму атаки для моделей 2 та 3 для великої кількості наявних замірів ключа з помилками. Оскільки шифр AES (як і багато інших симетричних шифрів) має складність O(1), то зі збільшенням розміру ключа не сильно збільшується складність виконання алгоритму. При цьому збільшення розміру ключа збільшує кількість необхідних статистичних замірів для успішної атаки сторонніми каналами. Але збільшення не сильно впливає на результуючу складність виконання атаки.

Подальші дослідження в цьому напрямку мають фокусуватися на досліджені більш спеціалізованих моделей атак що відбулися. Наприклад, моделі в яких в результаті атаки відомо вагу ключа, моделі в яких відомо окремі байти ключа (і, відповідно, різні розподіли ймовірностей помилок в бітах в залежності від значень відомих байтів).

ПЕРЕЛІК ПОСИЛАНЬ

1. Anderson R., Bond M., Clulow J., Skorobogatov, S. Cryptographic processors – a survey. — 17 pp. : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.1740&rep=rep1&type=pdf.

2. YongBin Zhou, DengGuo Feng. Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing —2005 —34 pp. : https://eprint.iacr.org/2005/388.

3. Randolph M., Diehl, W. Power side-channel attack analysis: A review of 20 years of study for the layman. Cryptography, 2020, 4.2: 15. : https://www.mdpi.com/2410-387X/4/2/15

4. Blömer J., Seifert J.-P., Fault based cryptanalysis of the advanced encryption standard (AES). In: Financial Cryptography: 7th International Conference, FC 2003, Guadeloupe, French West Indies, January 27-30, 2003. Revised Papers 7. Springer Berlin Heidelberg, 2003. p. 162-181. : https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=9cacdc50f8f0a5b107f45860908fa3c221254f1e

5. Neve, M., Tiri, K. On the complexity of side-channel attacks on AES-256–methodology and quantitative results on cache attacks. Cryptology ePrint Archive, 2007. : https://eprint.iacr.org/2007/318.pdf

6. Suresh, C. Towards sound approaches to counteract power-analysis attacks. In: Advances in Cryptology-CRYPTO'99. Springer-Verlag, 1999. p. 398-412. : https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=4fb1900092a86d1cdd6996cd95acfa339456dbf2

7. Kocher, P., Jaffe, J., Jun, B. Differential power analysis. In: Advances in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19. Springer Berlin Heidelberg, 1999. p. 388-397. https://www.rambus.com/wp-content/uploads/2015/08/DPA.pdf

8. Sweeney, P., Federal Information Processing Standards Publication

197. 2001. : https://csrc.nist.gov/files/pubs/fips/197/final/docs/fips-197.pdf

9. Robert, S. Permutation generation methods. pp150-151, 2008. https://homepage.divms.uiowa.edu/~goodman/22m150.dir/2007/Permutation%20Generation%20Methods.pdf

10. Гнєденко, Б.В. Курс теорії ймовірностей – К.: Видавничо-поліграфічний центр "Київський університет 2010. – 464 с https://probability.knu.ua/userfiles/yamnenko/Gnedenko.pdf

11. Толмачов, Є. Ю. Криптографічні атаки на RSA на основі інформації з побічного каналу : дипломна робота ... бакалавра : 113 Прикладна математика / Толмачов Євгеній Юрійович. – Київ, 2022. – 62 с. https://ela.kpi.ua/server/api/core/bitstreams/fd815484-343f-4652-96a7-846fac393838/content

ДОДАТОК А ТЕКСТИ ПРОГРАМ

А.1 Програма 1 - набір методів для підрахунку ймовірності успіху та складності для запропонованих моделей

```
import numpy as np
import math
from itertools import combinations


def p_model_1(p, N, k):
    '''
    функція що рахує ймовірність успіху для моделі 1
    '''
    return sum([math.comb(N,i)*(p**i)*((1-p)**(N-i))
                for i in range(k+1)])

def p_i(p, N, i):
    '''
    допоміжна функція що рахує параметр p_i
    '''
    return math.comb(N,i)*(p**i)*((1-p)**(N-i))

def complexity_mean_1(p, N, k):
    '''
    функція що рахує середню складність для моделі 1
    '''
    c = 0
    c = c + p_i(p, N, 0)
    for i in range(1, k+1):
```

```
        p_ = p_i(p, N, i)
        C = sum([math.comb(N, j) for j in range(i)])
        + 1/2*math.comb(N, i)
        c += p_*C
    p_greater_than_k = 1 - sum([p_i(p, N, i) for i in range(k+1)])
    C = sum([math.comb(N, i) for i in range(k+1)])
    c+=p_greater_than_k*C
    return c

def complexity_worst(p, N, k):
    '''
    Функція що обчислює складність алгоритму виправлення
    помилок для всіх моделей в найгіршому випадку
    '''
    c = 0
    for i in range(0, k+1):
        c+=math.comb(N, i)
    return c

def p_2(p, N, k, r):
    '''
    більш гарний варіант функції з минулої роботи що рахує ймовірність
    успіху для моделі 2
    '''
    pr = 0
    q_r = sum([math.comb(r,i)*p**(r-i)*(1-p)**(i)
                for i in range(int(r/2)+1)])
    # print(q_r)
    for i in range(k+1):
        pr += math.comb(N, i)*(q_r**i)*((1-q_r)**(N-i))
    return pr
```

```
def complexity_mean_2(p, N, k, r):
    '''
    більш гарний варіант функції з минулої роботи що
    рахує середню складність для моделі 2
    '''
    c = 0
    q = sum([math.comb(r,i)*p**(r-i)*(1-p)**(i)
                for i in range(int(r/2)+1)])
    c += x_i(q, 0, N, k)
    for i in range(k):
        x = x_i(q, i+1, N, k)
        C = sum([math.comb(N, j) for j in range(i)])
        + 1/2*math.comb(N, i)
        c += x*C
    x_greater_than_k = 1 - sum([x_i(q, i+1, N, k) for i in range(k+1)]
    C = sum([math.comb(N, i) for i in range(k+1)])
    c+=x_greater_than_k*C
    return c

def p_k_3(p, N, k):
    '''
    Функція що обчислює ймовірності успіху алгоритму
    виправлення помилок для моделі 3
    '''
    pr = 0
    q = q_r_3(p)
    for i in range(k+1):
        pr += math.comb(N, i)*(q**i)*((1-q)**(N-i))
    return pr
```

```
def q_r_3(p):
    '''
    Допоміжна функція що рахує параметр q_m для
    ймовірності успіху алгоритму виправлення помилок для моделі 3
    '''
    q = 0
    position = [i for i in range(len(p))]
    for i in range(int(len(p)/2)+1):
        for comb in combinations(position, i):
            q+=np.prod([1-pr if i in comb else pr
                        for i, pr in enumerate(p)])

    return q

def complexity_mean_3(p, N, k):
    '''
    функція що рахує середню складність для моделі 3
    '''
    c = 0
    q = q_r(p)
    c += x_i(q, 0, N, k)
    for i in range(k):
        x = x_i(q, i+1, N, k)
        C = sum([math.comb(N, j) for j in range(i)])
        + 1/2*math.comb(N, i)
        c += x*C
    x_greater_than_k = 1 - sum([x_i(q, i+1, N, k) for i in range(k+1)]
    C = sum([math.comb(N, i) for i in range(k+1)])
    c+=x_greater_than_k*C
    return c
```

```
def x_i(q_r, i, N, k):
    '''
    допоміжна функція що рахує параметр x_i
    '''
    return math.comb(N,i)*(q_r**i)*((1-q_r)**(N-i))

def q_r_limit(l, N, r):
    '''
    Функція що рахує апроксимований параметр q_m
    '''
    return 1 - sum([(float(l**i) * float(math.e**(-l)))/
    float(math.factorial(i)) for i in range(int(r/2)+1)])

def p_q_limit(p, N, k, r):
    '''
    Функція що рахує ймовірність успіху алгоритму
    моделі з апроксимованим параметром q_m
    '''
    pr = 0
    l = p*r
    q_r = q_r_limit(l, N, r)
    for i in range(k+1):
        pr += math.comb(N, i)*(q_r**i)*((1-q_r)**(N-i))
    return pr

def complexity_mean_q(p, N, k, r):
    '''
    Функція що рахує середню складність алгоритму
    моделі з апроксимованим параметром q_m
    '''
    c = 0
```

```
    l = p*r
    q = q_r_limit(l, N, r)
    c += x_i(q, 0, N, k)
    for i in range(k):
        x = x_i(q, i+1, N, k)
        C = sum([math.comb(N, j) for j in range(i)])
        + 1/2*math.comb(N, i)
        c += x*C
    x_greater_than_k = 1 - sum([x_i(q, i+1, N, k) for i in range(k+1)]
    C = sum([math.comb(N, i) for i in range(k+1)])
    c+=x_greater_than_k*C
    return c

# Параметри для обрахунків

k = [1, 2, 5, 10]
r_ = [3, 5, 7]
p_l = [np.array([0.01 * i for i in range(1,r+1)])
*(0.3/(0.01 * r)) for r in r_]
p_sqr = [np.array([0.01 * (i**2) for i in range(1,r+1)])
*(0.3/(0.01 * (r**2))) for r in r_]
p_pow = [np.array([0.01**(1/i) for i in range(1,r+1)])
*(0.3/(0.01**(1/r))) for r in r_]
N = [128, 192, 256]
p_all = [tuple(pr) for pr in p_sqr] +
[tuple(pr) for pr in p_l] +
[tuple(pr) for pr in p_pow]
p_classic = [0.01, 0.1, 0.15, 0.3]

# Обрахунок складності та ймовірностей
```

```
N_model_1_p =
for n in N:
    n_d =
    for pr in p_classic:
        n_d[pr] = [p_model_1(pr, n, e) for e in k]
    N_model_1_p[n] = n_d

N_model_1_cw =
for n in N:
    n_d = [complexity_worst(n,e) for e in k]
    N_model_1_cw[n] = n_d

N_model_1_m =
for n in N:
    n_d =
    for pr in p_classic:
        n_d[pr] = [complexity_mean_1(pr, n, e) for e in k]
    N_model_1_m[n] = n_d


N_model_2_p =
for n in N:
    n_d =
    for pr in p_classic:
        n_r =
        for r in [3, 5, 7]:
            n_r[r] = [p_2(pr, n, e, r) for e in k]
        n_d[pr] = n_r
    N_model_2_p[n] = n_d

N_model_2_m =
```

```
for n in N:
    n_d =
    for pr in p_classic:
        n_r =
        for r in [3, 5, 7]:
            n_r[r] = [complexity_mean_2(pr, n, e, r) for e in k]
        n_d[pr] = n_r
    N_model_2_m[n] = n_d

N_model_3_p =
for n in N:
    n_d =
    for pr in p_all:
        n_d[pr] = [p_k(pr, n, e) for e in k]
    N_model_3_p[n] = n_d

N_model_3_m =
for n in N:
    n_d =
    for pr in p_all:
        n_d[pr] = [complexity_mean_3(pr, n, e) for e in k]
    N_model_3_m[n] = n_d

q_limit =
for n in N:
    n_d =
    for pr in [0.3, 0.4, 0.45]:
        n_r =
        for r in [25, 51, 101, 201]:
            l = pr*r
            n_r[r] = q_r_limit(l, N, r)
```

```
        n_d[pr] = n_r
    q_limit[n] = n_d

N_model_q_p =
for n in N:
    n_d =
    for pr in [0.3, 0.4, 0.45]:
        n_r =
        for r in [25, 51, 101, 201]:
            n_r[r] = [p_q_limit(pr, n, e, r) for e in k]
        n_d[pr] = n_r
    N_model_q_p[n] = n_d


N_model_q_m =
for n in N:
    n_d =
    for pr in [0.3, 0.4, 0.45]:
        n_r =
        for r in [25, 51, 101, 201]:
            n_r[r] = [complexity_mean_q(pr, n, e, r) for e in k]
        n_d[pr] = n_r
    N_model_q_m[n] = n_d

# Створення файлу з таблицями

with open('tables.txt', 'w') as f:
    for n in N:
        f.write(" \\multicolumn{5}{c}{$ \\beta =" + str(n)
        + "$} \\\\ \n")
        f.write(" \\midrule  \n")
```

```
    f.write("k = ")
    for e in k:
        f.write( " & " + f'$e$')
    f.write("\\\\ \n  \\midrule  \n")
    f.write("$C^k_ \\beta$")
    for c in N_model_1_cw[n]:
        two_to_power = "2^{"+f'{np.around(math.log2(c), 2)}'+"}"
        f.write( " & " + f'$two_to_power$')
    f.write("\\\\ \n  \\midrule  \n")  #swap hline for midrule

f.write("\\\\ \n  \\bottomrule  \n")  #swap hline for midrule

f.write(' \n  \n')
for n in N:
    f.write(" \\multicolumn{5}{c}{$ \\beta =" + str(n) +
    "$} \\\\ \n")
    f.write(" \\midrule  \n")
    for pr in p_all:
        f.write(" \\multicolumn{5}{c}{$m = "+ str(len(pr)) +
        ", \\quad p =" + str([np.around(p, 2) for p in pr]) + ", \
        f.write(" \\midrule  \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')
        f.write("\\\\ \n  \\midrule  \n")
        f.write("$C^k_ \\beta$")
        for c in N_model_3_m[n][pr]:
            two_to_power = "2^{"+f'{np.around(math.log2(c),
            2)}'+"}"
            f.write( " & " + f'$two_to_power$')
        f.write("\\\\ \n  \\midrule  \n")
```

```
f.write(' \n  \n')

for n in N:
    f.write(" \\multicolumn{5}{c}{$ \\beta =" + str(n) +
    "$} \\\\ \n")
    f.write(" \\midrule  \n")
    for pr in p_all:
        f.write(" \\multicolumn{5}{c}
        {$m = "+ str(len(pr)) + ", \\quad p =" + str([np.around(p,
        f.write(" \\midrule  \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')
        f.write("\\\\ \n  \\midrule  \n")
        f.write("$C^k_ \\beta$")
        for c in N_model_3_p[n][pr]:
            if c<0.00001:
                two_to_power = "2^{"+f'{np.around(math.log2(c),
                2)}'+"}"
            else:
                two_to_power = f'{np.around(c, 4)}'
            f.write( " & " + f'$two_to_power$')
        f.write("\\\\ \n  \\midrule  \n")

f.write(' \n  \n')

for n in N:
    f.write(" \\multicolumn{5}{c}{$ \\beta =" + str(n)
    + "$} \\\\ \n")
    f.write(" \\midrule  \n")
```

```
    f.write("k = ")
    for e in k:
        f.write( " & " + f'$e$')
    f.write("\\\\ \n  \\midrule  \n")
    for pr in p_classic:
        f.write(f'$p = pr$')
        for c in N_model_1_p[n][pr]:
            if c<0.00001:
                two_to_power = "2^{"+f'{np.around(math.log2(c),
                2)}'+"}"
            else:
                two_to_power = f'{np.around(c, 4)}'
            f.write( " & " + f'$two_to_power$')
        f.write("\\\\ \n  \\midrule  \n")

for n in N:
    f.write(" \\multicolumn{5}{c}{$ \\beta =" + str(n) +
    "$} \\\\ \n")
    f.write(" \\midrule  \n")
    f.write("k = ")
    for e in k:
        f.write( " & " + f'$e$')
    f.write("\\\\ \n  \\midrule  \n")
    for pr in p_classic:
        f.write(f'$p = pr$')
        for c in N_model_1_m[n][pr]:
            if c>0.00001:
                two_to_power = "2^{"+f'{np.around(math.log2(c),
                2)}'+"}"
            else:
                two_to_power = f'{np.around(c, 4)}'
```

```
            f.write( " & " + f'$two_to_power$')
        f.write("\\\\ \n  \\midrule  \n")

f.write("\\\\ \n  \\bottomrule  \n")  #swap hline for midrule
    for n in N:
    f.write("\\multicolumn5c$\\beta =" + str(n) + "$ \\\\ \n")
    f.write("\\midrule \n")
    for r in r_:
        f.write("\\multicolumn5c$r =" + str(r) +" $ \\\\ \n")
        f.write("\\midrule \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')

        f.write("\\\\ \n \\midrule \n")
        for pr in p_classic:
            f.write(f'$p = pr$')
            for c in N_model_2_p[n][pr][r]:
                if c<0.00001:
                    two_to_power = "2^"+f'np.around(math.log2(c),
                    2)'+""
                else:
                    two_to_power = f'np.around(c, 4)'
                f.write( " & " + f'$two_to_power$')

            f.write("\\\\ \n \\midrule \n")

f.write('\n \n')

for n in N:
    f.write("\\multicolumn5c$\\beta =" + str(n) + "$ \\\\ \n")
```

```
    f.write("\\midrule \n")
    for r in r_:
        f.write("\\multicolumn5c$r =" + str(r) +" $ \\\\ \n")
        f.write("\\midrule \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')

        f.write("\\\\ \n \\midrule \n")
        for pr in p_classic:

            f.write(f'$p = pr$')
            for c in N_model_2_m[n][pr][r]:
                two_to_power = "2^"+f'np.around(math.log2(c),
                2)'+""
                f.write( " & " + f'$two_to_power$')
            f.write("\\\\ \n \\midrule \n")

f.write('\n \n')
for n in N[0:1]:
    f.write("\\multicolumn5c$\\beta =" + str(n) + "$ \\ \n")
    f.write("\\midrule \n")
    for pr in [0.3, 0.4, 0.45]:
        for r in [25, 51, 101, 201]:
            f.write( " & " + f'$r$')
        f.write(f'$p = pr$')
        for c in [q_limit[n][pr][k] for k in
        q_limit[n][pr].keys()]:
            if c<0.00001:
                two_to_power = "2^"+f'np.around(math.log2(c),
                2)'+""
```

```
            else:
                two_to_power = f'np.around(c, 4)'
            f.write( " & " + f'$two_to_power$')
        f.write("\\ \n \\midrule \n")
f.write('\n \n')

for n in N:
    f.write("\\multicolumn5c$\\beta =" + str(n) + "$ \\ \n")
    f.write("\\midrule \n")
    for r in [25, 51, 101, 201]:
        f.write("\\multicolumn5c$r =" + str(r) +" $ \\ \n")
        f.write("\\midrule \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')

        f.write("\\ \n \\midrule \n")
        for pr in [0.3, 0.4, 0.45]:

            f.write(f'$p = pr$')
            for c in N_model_q_p[n][pr][r]:
                if c<0.00001:
                    two_to_power = "2^"+f'np.around(math.log2(c),
                    2)'+""
                else:
                    two_to_power = f'np.around(c, 4)'
                f.write( " & " + f'$two_to_power$')
            f.write("\\ \n \\midrule \n")

f.write('\n \n')
```

```
for n in N:
    f.write("\\multicolumn5c$\\beta =" + str(n) + "$ \\\\ \n")
    f.write("\\midrule \n")
    for r in [25, 51, 101, 201]:
        f.write("\\multicolumn5c$r =" + str(r) +" $ \\\\ \n")
        f.write("\\midrule \n")
        f.write("k = ")
        for e in k:
            f.write( " & " + f'$e$')

        f.write("\\\\ \n \\midrule \n")
        for pr in [0.3, 0.4, 0.45]:

            f.write(f'$p = pr$')
            for c in N_model_q_m[n][pr][r]:
                two_to_power = "2^"+f'np.around(math.log2(c),
                2)'+""
            f.write( " & " + f'$two_to_power$')
            f.write("\\\\ \n \\midrule \n")
```

ДОДАТОК Б ВЕЛИКІ РИСУНКИ ТА ТАБЛИЦІ

Таблиця Б.1 – Ймовірність успіху атаки в моделі 2 при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

$\beta = 128$				
$m = 3$				
k =	1	2	5	10
$p = 0.01$	0.9993	1.0	1.0	1.0
$p = 0.1$	0.1237	0.3016	0.8491	0.999
$p = 0.15$	0.003	0.0142	0.204	0.8441
$p = 0.3$	$2^{-39.76}$	$2^{-35.59}$	$2^{-26.04}$	0.0
$m = 5$				
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.7005	0.9021	0.9991	1.0
$p = 0.15$	0.1425	0.3349	0.8725	0.9993
$p = 0.3$	$2^{-28.18}$	$2^{-24.49}$	0.0	0.0038
$m = 7$				
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.9517	0.9946	1.0	1.0
$p = 0.15$	0.5404	0.7971	0.9951	1.0
$p = 0.3$	$2^{-20.59}$	$2^{-17.32}$	0.0007	0.0603
$\beta = 192$				
$m = 3$				
k =	1	2	5	10

$p = 0.01$	0.9984	1.0	1.0	1.0
$p = 0.1$	0.028	0.0932	0.5494	0.9798
$p = 0.15$	0.0001	0.0005	0.022	0.3776
$p = 0.3$	$2^{-61.65}$	$2^{-56.91}$	$2^{-45.62}$	$2^{-31.98}$
		$m = 5$		
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.5101	0.7725	0.9934	1.0
$p = 0.15$	0.0352	0.1124	0.5965	0.9855
$p = 0.3$	$2^{-44.05}$	$2^{-39.79}$	$2^{-29.96}$	$2^{-18.74}$
		$m = 7$		
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.9027	0.9839	1.0	1.0
$p = 0.15$	0.3236	0.5892	0.9696	1.0
$p = 0.3$	$2^{-32.47}$	$2^{-28.64}$	$2^{-20.06}$	0.0005
		$\beta = 256$		
		$m = 3$		
k =	1	2	5	10
$p = 0.01$	0.9972	0.9999	1.0	1.0
$p = 0.1$	0.0058	0.0247	0.276	0.8922
$p = 0.15$	$2^{-19.01}$	0.0	0.0015	0.0871
$p = 0.3$	$2^{-83.71}$	$2^{-78.56}$	$2^{-66.03}$	$2^{-50.3}$
		$m = 5$		
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.3554	0.6248	0.9761	1.0

$p = 0.15$	0.008	0.0325	0.322	0.917
$p = 0.3$	$2^{-60.08}$	$2^{-55.42}$	$2^{-44.36}$	$2^{-31.07}$
		$m = 7$		
k =	1	2	5	10
$p = 0.01$	1.0	1.0	1.0	1.0
$p = 0.1$	0.8449	0.9663	0.9999	1.0
$p = 0.15$	0.1831	0.4001	0.9071	0.9997
$p = 0.3$	$2^{-44.51}$	$2^{-40.27}$	$2^{-30.48}$	$2^{-19.28}$

Таблиця Б.2 – Складність атаки для моделі 2 в середньому при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

		$\beta = 128$		
		$m = 3$		
k =	1	2	5	10
$p = 0.01$	$2^{6.97}$	$2^{12.96}$	$2^{27.98}$	$2^{47.76}$
$p = 0.1$	$2^{6.55}$	$2^{12.04}$	$2^{24.69}$	$2^{42.59}$
$p = 0.15$	$2^{6.99}$	$2^{12.95}$	$2^{27.46}$	$2^{44.41}$
$p = 0.3$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
		$m = 5$		
k =	1	2	5	10
$p = 0.01$	$2^{7.02}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$p = 0.1$	$2^{5.81}$	$2^{11.53}$	$2^{26.45}$	$2^{46.23}$
$p = 0.15$	$2^{6.49}$	$2^{11.94}$	$2^{24.57}$	$2^{42.85}$
$p = 0.3$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.8}$

$m = 7$				
k =	1	2	5	10
$p = 0.01$	$2^{7.02}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$p = 0.1$	$2^{6.53}$	$2^{12.51}$	$2^{27.53}$	$2^{47.31}$
$p = 0.15$	$2^{5.75}$	$2^{11.19}$	$2^{25.8}$	$2^{45.57}$
$p = 0.3$	$2^{7.01}$	$2^{13.01}$	$2^{28.03}$	$2^{47.66}$
$\beta = 192$				
$m = 3$				
k =	1	2	5	10
$p = 0.01$	$2^{7.52}$	$2^{14.1}$	$2^{30.9}$	$2^{53.71}$
$p = 0.1$	$2^{7.46}$	$2^{13.84}$	$2^{29.24}$	$2^{47.58}$
$p = 0.15$	$2^{7.59}$	$2^{14.17}$	$2^{30.91}$	$2^{52.81}$
$p = 0.3$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$m = 5$				
k =	1	2	5	10
$p = 0.01$	$2^{7.6}$	$2^{14.17}$	$2^{30.98}$	$2^{53.79}$
$p = 0.1$	$2^{6.35}$	$2^{12.33}$	$2^{28.61}$	$2^{51.41}$
$p = 0.15$	$2^{7.43}$	$2^{13.78}$	$2^{29.04}$	$2^{47.41}$
$p = 0.3$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$m = 7$				
k =	1	2	5	10
$p = 0.01$	$2^{7.6}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$p = 0.1$	$2^{6.88}$	$2^{13.43}$	$2^{30.22}$	$2^{53.04}$
$p = 0.15$	$2^{6.62}$	$2^{12.45}$	$2^{27.75}$	$2^{50.42}$
$p = 0.3$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.79}$
$\beta = 256$				
$m = 3$				

k =	1	2	5	10
$p = 0.01$	$2^{7.9}$	$2^{14.9}$	$2^{32.96}$	$2^{57.9}$
$p = 0.1$	$2^{7.97}$	$2^{14.9}$	$2^{32.28}$	$2^{53.98}$
$p = 0.15$	$2^{8.01}$	$2^{15.01}$	$2^{33.06}$	$2^{57.79}$
$p = 0.3$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$m = 5$				
k =	1	2	5	10
$p = 0.01$	$2^{8.01}$	$2^{15.0}$	$2^{33.06}$	$2^{58.01}$
$p = 0.1$	$2^{6.97}$	$2^{13.22}$	$2^{29.98}$	$2^{54.84}$
$p = 0.15$	$2^{7.96}$	$2^{14.87}$	$2^{32.14}$	$2^{53.55}$
$p = 0.3$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$m = 7$				
k =	1	2	5	10
$p = 0.01$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$p = 0.1$	$2^{7.1}$	$2^{14.01}$	$2^{32.06}$	$2^{57.0}$
$p = 0.15$	$2^{7.37}$	$2^{13.75}$	$2^{29.48}$	$2^{53.52}$
$p = 0.3$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$

Таблиця Б.3 – Ймовірність успіху атаки в моделі 3 при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

$\beta = 128$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				
k =	1	2	5	10
Pr	0.0089	0.0357	0.3447	0.9311

$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
Pr	0.5482	0.803	0.9955	1.0
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
Pr	0.9439	0.9932	1.0	1.0
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
Pr	0.0	0.0002	0.0112	0.2804
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
Pr	0.0402	0.1258	0.6287	0.9892
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
Pr	0.3901	0.6627	0.9824	1.0
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
Pr	0.0159	0.0586	0.4449	0.962
$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				
k =	1	2	5	10
Pr	0.2237	0.4617	0.935	0.9999
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
Pr	0.5028	0.767	0.9932	1.0
$\beta = 192$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				
k =	1	2	5	10

Pr	0.0004	0.0024	0.0643	0.5906
$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
Pr	0.3316	0.5985	0.9715	1.0
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
Pr	0.8879	0.9799	1.0	1.0
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
Pr	$2^{-24.12}$	$2^{-20.67}$	0.0001	0.0158
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
Pr	0.0046	0.0203	0.248	0.8755
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
Pr	0.1856	0.4043	0.9098	0.9997
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
Pr	0.0011	0.0056	0.1116	0.7123
$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				
k =	1	2	5	10
Pr	0.0736	0.2024	0.7488	0.9961
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
Pr	0.2858	0.5436	0.9592	0.9999
$\beta = 256$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				

k =	1	2	5	10
Pr	0.0	0.0001	0.0078	0.2234
$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
Pr	0.1898	0.4103	0.912	0.9997
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
Pr	0.8229	0.9583	0.9999	1.0
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
Pr	$2^{-33.24}$	$2^{-29.4}$	$2^{-20.79}$	0.0003
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
Pr	0.0005	0.0027	0.0689	0.6012
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
Pr	0.0827	0.2211	0.7701	0.9969
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
Pr	0.0001	0.0004	0.0187	0.3462
$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				
k =	1	2	5	10
Pr	0.0224	0.0775	0.5028	0.9719
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
Pr	0.1532	0.3517	0.8806	0.9994

Таблиця Б.4 – Складність атаки для моделі 3 в середньому при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

$\beta = 128$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				
k =	1	2	5	10
C_β^k	$2^{6.96}$	$2^{12.87}$	$2^{27.04}$	$2^{43.11}$
$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
C_β^k	$2^{5.74}$	$2^{11.2}$	$2^{25.83}$	$2^{45.6}$
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
C_β^k	$2^{6.49}$	$2^{12.46}$	$2^{27.49}$	$2^{47.26}$
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
C_β^k	$2^{7.01}$	$2^{13.01}$	$2^{28.0}$	$2^{47.11}$
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
C_β^k	$2^{6.83}$	$2^{12.57}$	$2^{25.95}$	$2^{41.36}$
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
C_β^k	$2^{5.91}$	$2^{11.17}$	$2^{25.12}$	$2^{44.83}$
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
C_β^k	$2^{6.93}$	$2^{12.79}$	$2^{26.7}$	$2^{42.26}$
$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				
k =	1	2	5	10

C_β^k	$2^{6.27}$	$2^{11.59}$	$2^{24.45}$	$2^{43.7}$
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
C_β^k	$2^{5.77}$	$2^{11.15}$	$2^{25.64}$	$2^{45.4}$
$\beta = 192$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				
k =	1	2	5	10
C_β^k	$2^{7.59}$	$2^{14.16}$	$2^{30.79}$	$2^{52.04}$
$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
C_β^k	$2^{6.6}$	$2^{12.44}$	$2^{27.79}$	$2^{50.47}$
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
C_β^k	$2^{6.83}$	$2^{13.36}$	$2^{30.16}$	$2^{52.97}$
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
C_β^k	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.75}$
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
C_β^k	$2^{7.56}$	$2^{14.09}$	$2^{30.27}$	$2^{50.01}$
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
C_β^k	$2^{6.95}$	$2^{12.91}$	$2^{27.39}$	$2^{49.32}$
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
C_β^k	$2^{7.58}$	$2^{14.15}$	$2^{30.66}$	$2^{51.42}$
$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				

k =	1	2	5	10
C_β^k	$2^{7.29}$	$2^{13.51}$	$2^{28.25}$	$2^{47.76}$
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
C_β^k	$2^{6.7}$	$2^{12.55}$	$2^{27.59}$	$2^{50.17}$
$\beta = 256$				
$m = 3, \quad p = [0.03, 0.13, 0.3], \quad p_{mean} = 0.1556$				
k =	1	2	5	10
C_β^k	$2^{8.01}$	$2^{15.0}$	$2^{33.04}$	$2^{57.45}$
$m = 5, \quad p = [0.01, 0.05, 0.11, 0.19, 0.3], \quad p_{mean} = 0.132$				
k =	1	2	5	10
C_β^k	$2^{7.35}$	$2^{13.73}$	$2^{29.47}$	$2^{53.59}$
$m = 7, \quad p = [0.01, 0.02, 0.06, 0.1, 0.15, 0.22, 0.3], \quad p_{mean} = 0.1224$				
k =	1	2	5	10
C_β^k	$2^{7.04}$	$2^{13.93}$	$2^{31.97}$	$2^{56.91}$
$m = 3, \quad p = [0.1, 0.2, 0.3], \quad p_{mean} = 0.2$				
k =	1	2	5	10
C_β^k	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$m = 5, \quad p = [0.06, 0.12, 0.18, 0.24, 0.3], \quad p_{mean} = 0.18$				
k =	1	2	5	10
C_β^k	$2^{8.0}$	$2^{14.99}$	$2^{32.86}$	$2^{56.21}$
$m = 7, \quad p = [0.04, 0.09, 0.13, 0.17, 0.21, 0.26, 0.3], \quad p_{mean} = 0.1714$				
k =	1	2	5	10
C_β^k	$2^{7.67}$	$2^{14.28}$	$2^{30.2}$	$2^{52.14}$
$m = 3, \quad p = [0.01, 0.14, 0.3], \quad p_{mean} = 0.1511$				
k =	1	2	5	10
C_β^k	$2^{8.0}$	$2^{15.0}$	$2^{33.0}$	$2^{57.12}$

$m = 5, \quad p = [0.01, 0.08, 0.16, 0.24, 0.3], \quad p_{mean} = 0.1567$				
k =	1	2	5	10
C_β^k	$2^{7.89}$	$2^{14.72}$	$2^{31.52}$	$2^{52.07}$
$m = 7, \quad p = [0.01, 0.06, 0.12, 0.18, 0.23, 0.27, 0.3], \quad p_{mean} = 0.1673$				
k =	1	2	5	10
C_β^k	$2^{7.46}$	$2^{13.89}$	$2^{29.56}$	$2^{53.18}$

Таблиця Б.5 – Апроксимація ймовірності успіху алгоритмів 3.2, 3.3 при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

$\beta = 128$				
$m = 25$				
k =	1	2	5	10
$p = 0.3$	0.0253	0.0861	0.5339	0.9785
$p = 0.4$	$2^{-38.05}$	$2^{-33.94}$	$2^{-24.58}$	0.0001
$p = 0.45$	$2^{-70.38}$	$2^{-65.33}$	$2^{-53.16}$	$2^{-38.13}$
$m = 51$				
k =	1	2	5	10
$p = 0.3$	0.7347	0.9199	0.9994	1.0
$p = 0.4$	$2^{-21.57}$	$2^{-18.23}$	0.0004	0.0438
$p = 0.45$	$2^{-57.18}$	$2^{-52.46}$	$2^{-41.29}$	$2^{-27.9}$
$m = 101$				
k =	1	2	5	10
$p = 0.3$	0.9989	1.0	1.0	1.0
$p = 0.4$	0.0033	0.0153	0.2129	0.8521

$p = 0.45$	$2^{-41.49}$	$2^{-37.26}$	$2^{-27.52}$	0.0
		$m = 201$		
k =	1	2	5	10
$p = 0.3$	1.0	1.0	1.0	1.0
$p = 0.4$	0.4325	0.7046	0.9876	1.0
$p = 0.45$	$2^{-24.56}$	$2^{-21.05}$	0.0001	0.0153
		$\beta = 192$		
		$m = 25$		
k =	1	2	5	10
$p = 0.3$	0.0022	0.0106	0.1684	0.8005
$p = 0.4$	$2^{-59.06}$	$2^{-54.38}$	$2^{-43.28}$	$2^{-29.96}$
$p = 0.45$	$2^{-108.02}$	$2^{-102.39}$	$2^{-88.45}$	$2^{-70.42}$
		$m = 51$		
k =	1	2	5	10
$p = 0.3$	0.5557	0.8082	0.9957	1.0
$p = 0.4$	$2^{-33.96}$	$2^{-30.06}$	$2^{-21.31}$	0.0003
$p = 0.45$	$2^{-88.06}$	$2^{-82.76}$	$2^{-69.83}$	$2^{-53.45}$
		$m = 101$		
k =	1	2	5	10
$p = 0.3$	0.9976	0.9999	1.0	1.0
$p = 0.4$	0.0001	0.0006	0.0239	0.3926
$p = 0.45$	$2^{-64.28}$	$2^{-59.48}$	$2^{-48.0}$	$2^{-34.05}$
		$m = 201$		
k =	1	2	5	10
$p = 0.3$	1.0	1.0	1.0	1.0
$p = 0.4$	0.221	0.457	0.9322	0.9998
$p = 0.45$	$2^{-38.53}$	$2^{-34.46}$	$2^{-25.19}$	0.0

$\beta = 256$				
$m = 25$				
k =	1	2	5	10
$p = 0.3$	0.0002	0.0011	0.0364	0.4668
$p = 0.4$	$2^{-80.24}$	$2^{-75.15}$	$2^{-62.81}$	$2^{-47.39}$
$p = 0.45$	$2^{-145.83}$	$2^{-139.78}$	$2^{-124.6}$	$2^{-104.44}$
$m = 51$				
k =	1	2	5	10
$p = 0.3$	0.4033	0.6751	0.9836	1.0
$p = 0.4$	$2^{-46.51}$	$2^{-42.21}$	$2^{-32.24}$	$2^{-20.74}$
$p = 0.45$	$2^{-119.1}$	$2^{-113.39}$	$2^{-99.21}$	$2^{-80.72}$
$m = 101$				
k =	1	2	5	10
$p = 0.3$	0.9958	0.9999	1.0	1.0
$p = 0.4$	$2^{-18.76}$	0.0	0.0017	0.0942
$p = 0.45$	$2^{-87.24}$	$2^{-82.02}$	$2^{-69.31}$	$2^{-53.26}$
$m = 201$				
k =	1	2	5	10
$p = 0.3$	1.0	1.0	1.0	1.0
$p = 0.4$	0.106	0.2675	0.8176	0.9983
$p = 0.45$	$2^{-52.66}$	$2^{-48.18}$	$2^{-37.69}$	$2^{-25.33}$

Таблиця Б.6 – Апроксимація складності атаки в середньому алгоритмів 3.2, 3.3 при розмірі ключа β, наявності m вимірів для ключа з ймовірністю помилок p та k виправлень

$\beta = 128$				
$r = 25$				
k =	1	2	5	10
$p = 0.3$	$2^{6.89}$	$2^{12.7}$	$2^{26.36}$	$2^{41.66}$
$p = 0.4$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$p = 0.45$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$r = 51$				
k =	1	2	5	10
$p = 0.3$	$2^{5.86}$	$2^{11.64}$	$2^{26.59}$	$2^{46.36}$
$p = 0.4$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.7}$
$p = 0.45$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$r = 101$				
k =	1	2	5	10
$p = 0.3$	$2^{6.95}$	$2^{12.94}$	$2^{27.97}$	$2^{47.75}$
$p = 0.4$	$2^{6.99}$	$2^{12.94}$	$2^{27.43}$	$2^{44.33}$
$p = 0.45$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$r = 201$				
k =	1	2	5	10
$p = 0.3$	$2^{7.02}$	$2^{13.01}$	$2^{28.04}$	$2^{47.81}$
$p = 0.4$	$2^{5.84}$	$2^{11.14}$	$2^{25.32}$	$2^{45.06}$
$p = 0.45$	$2^{7.01}$	$2^{13.01}$	$2^{28.04}$	$2^{47.77}$
$\beta = 192$				
$r = 25$				
k =	1	2	5	10

$p = 0.3$	$2^{7.58}$	$2^{14.13}$	$2^{30.5}$	$2^{50.79}$
$p = 0.4$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
$p = 0.45$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
		$r = 51$		
k =	1	2	5	10
$p = 0.3$	$2^{6.32}$	$2^{12.38}$	$2^{28.81}$	$2^{51.62}$
$p = 0.4$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.79}$
$p = 0.45$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
		$r = 101$		
k =	1	2	5	10
$p = 0.3$	$2^{7.5}$	$2^{14.07}$	$2^{30.88}$	$2^{53.69}$
$p = 0.4$	$2^{7.59}$	$2^{14.17}$	$2^{30.9}$	$2^{52.76}$
$p = 0.45$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
		$r = 201$		
k =	1	2	5	10
$p = 0.3$	$2^{7.6}$	$2^{14.18}$	$2^{30.98}$	$2^{53.79}$
$p = 0.4$	$2^{6.86}$	$2^{12.77}$	$2^{27.4}$	$2^{49.66}$
$p = 0.45$	$2^{7.59}$	$2^{14.18}$	$2^{30.98}$	$2^{53.8}$
		$\beta = 256$		
		$r = 25$		
k =	1	2	5	10
$p = 0.3$	$2^{8.0}$	$2^{15.0}$	$2^{32.95}$	$2^{56.74}$
$p = 0.4$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$p = 0.45$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
		$r = 51$		
k =	1	2	5	10
$p = 0.3$	$2^{6.88}$	$2^{13.16}$	$2^{30.21}$	$2^{55.11}$

$p = 0.4$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
$p = 0.45$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
		$r = 101$		
k =	1	2	5	10
$p = 0.3$	$2^{7.87}$	$2^{14.87}$	$2^{32.93}$	$2^{57.87}$
$p = 0.4$	$2^{8.01}$	$2^{15.01}$	$2^{33.06}$	$2^{57.77}$
$p = 0.45$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$
		$r = 201$		
k =	1	2	5	10
$p = 0.3$	$2^{8.01}$	$2^{15.01}$	$2^{33.06}$	$2^{58.01}$
$p = 0.4$	$2^{7.6}$	$2^{14.14}$	$2^{29.91}$	$2^{52.53}$
$p = 0.45$	$2^{8.01}$	$2^{15.01}$	$2^{33.07}$	$2^{58.01}$