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

«На правах рукопису»
УДК (впишіть правильний УДК!)

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

Дипломна робота

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

зі спеціальності: 113 Прикладна математика
на тему: **«Побудова диференціальних атак на ARX-криптосистеми на прикладі модифікованого шифру Кипарис»**

Виконав:
студент IV курсу, групи ФІ-04
Сковрон Роман Васильович _____

Керівник:
доцент кафедри ММЗІ, к.т.н
Яковлєв Сергій Володимирович _____

Рецензент:
посада, степінь, звання
Прізвище Ім'я По-батькові _____

Засвідчую, що у цій дипломній роботі 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 — перший (бакалаврський)
Спеціальність — 113 Прикладна математика,
ОПП «Математичні методи криптографічного захисту інформації»

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

ЗАВДАННЯ
на дипломну роботу

Студент: Сковрон Роман Васильович

1. Тема роботи: «*Побудова диференціальних атак на ARX-криптосистеми на прикладі модифікованого шифру Кипарис*», науковий керівник дисертації: доцент кафедри ММЗІ, к.т.н Яковлєв Сергій Володимирович,

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

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

3. Об'єкт дослідження: інформаційні процеси в системах захисту інформації

4. Предмет дослідження: методи диференціального криптоаналізу для малоресурсних ширфів

5. Перелік завдань: провести огляд опублікованих джерел за тематикою дослідження, проаналізувати метод Зустріч у Фільтрі для криптоаналізу малоресурсних шифрів , запропонувати алгоритми побудови диференціальних характеристик і фільтра для малоресурсного шифра «Кипарис», проаналізувати застосовність методу Зустрічі у Фільтрі для модифікованого шифру «Кипарис»

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

7. Орієнтовний перелік публікацій: «планується доповідь на всеукраїнській конференції»

8. Дата видачі завдання: 10 вересня 2023 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2023 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2023 р.	Виконано
4	Огляд методу Зустріч у Фільтрі та його застосування до $CHAM$ та $SPECK$	Листопад-грудень 2023 р.	Виконано
5	Побудова алгоритму пошуку диференціальних характеристик для шифру «Кипарис»	Січень 2024 р.	Виконано
6	Запропоновано застосування метода Зустріч у Фільтрі	Березнь 2024 р.	Виконано
7	Проаналізовано отримані результати	Квітень 2024 р.	Виконано
8	Оформлення дипломної роботи	Травень 2024 р.	Виконано

Студент _____ Роман Сковрон

Керівник _____ Сергій Яковлєв

РЕФЕРАТ

Кваліфікаційна робота містить: 51 стор., 14 рисунки, 8 таблиць, 8 джерел.

У роботі вперше проаналізовано застосовність метода диференціального криптоаналізу на блоковий малоресурсний шифр «Кипарис» Зустріч у Фільтрі. Запропоновано алгоритм за побудови дифренцальних характеристик для модифікованого шифру «Кипарис». Застосовано метод Зустріч у Фільтрі до модифікованого шифру «Кипарис» за результатами якого проаналізовано застосовність цього метода. Крім того детально проаналізовано алгоритм пошуку всіх оптимальних диференціалів, виявлено його особливості роботи та його не повну коректність.

Метою дослідження є уточнення методів криптоаналізу малоресурсних криптосистем.

Об'єктом дослідження є інформаційні процеси в системах захисту інформації.

Предметом дослідження є методи диференціального криптоаналізу для малоресурсних шифрів.

СИМЕТРИЧНА КРИПТОГРАФІЯ, ДИФЕРЕНЦІАЛЬНИЙ КРИПТОАНАЛІЗ, ARX, «КИПАРИС», ЗУСТРІЧ У ФІЛЬТРІ

ABSTRACT

The thesis contains: 51 pages, 14 figures, 8 tables, 8 sources

In this thesis, for the first time analyzes the applicability of the method of differential cryptanalysis Meet in the Filter to the lightweight block cipher «Cypress». An algorithm for constructing differential characteristics is proposed. The method of Meeting in the Filter is applied to the modified cipher «Cypress», and the applicability of this method is analyzed. In addition, the algorithm for finding all optimal differentials is analyzed in detail, its peculiarities of work and its incomplete correctness are revealed.

The purpose of the thesis is to refine the methods of cryptanalysis of lightweight cryptosystems.

The research object are information processes in information security systems.

The research subject is differential cryptanalysis methods for lightweight ciphers.

SYMMETRIC CRYPTOGRAPHY, DIFFERENTIAL CRYPTANALYSIS, ARX, «CYPRESS», MEET IN THE FILTER

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

$\oplus$, XOR — операція плбітового додавання

$+$, ADD — додавання за модулем n

$\ll$ – побітовий не циклічний зсув вліво

$\lll$ — побітовий циклічний зсув вліво

$\gg$ — побітовий не циклічний зсув вправо

$\ggg$ — побітовий циклічний зсув вправо

xdp^+ — XOR диференціальна ймовірність через додавання (XOR differential probability of addition)

α_i, γ_i, β_i — $s-$бітове слово

ВСТУП

Актуальність дослідження. На сьогодній момент прослідковується стала тенденація до збільшення обсягу поширення інформаційних технологія у всіх сферах діяльності людини. В тому числі це відбувається за рахунок зростання кількості малопотужних пристроїв таких як смарткарток, пристрої інтернет речей, сенсорних мереж та інших подбіних малоресурсних пристроїв основною задачей яких є збір, обробка та передача через мережу інтернет. Саме тому необхідно постійно вдосконалювати методи шифрування на всіх можливих рівнях. ARX-криптосистеми є одні з найбільш вдалих рішень для малоресурсних систем, один з таких прикладів в Україні є малоресурсний шифр «Кипарис». Диференціальний аналіз є одни з основних методів криптоаналізу і отже актульність полягає в перевірці стійкості шифру «Кипарси» до методів диференціального криптоаналізу.

Метою дослідження є уточнення методів криптоаналізу малоресурсних криптосистем. Для досягнення мети необхідно виконати такі **завдання дослідження**

1) провести огляд опублікованих джерел за тематикою дослідження;

2) проаналізувати метод Зустріч у Фільтрі для криптоаналізу малоресурсних шифрів;

3) запропонувати алгоритми побудови диференціальних характеристик і фільтра для малоресурсного шифра «Кипарис»;

4) проаналізувати застосовність методу Зустрічі у Фільтрі для модифікованого шифру «Кипарис».

Об'єктом дослідження є інформаційні процеси в системах захисту інформації.

Предметом дослідження є методи диференціального криптоаналізу для малоресурсних шифрів.

При розв'язанні поставлених завдань використовувались такі

методи дослідження: методи абстрактної алгебри, теорії імовірностей, математичної статистики, комбінаторного аналізу, дискретної математики, теорії кодування, теорії складності алгоритмів, методи комп'ютерного та статистичного моделювання.

Наукова новизна отриманих результатів: у роботі вперше проаналізована застосовність методу Зустірч у Фільтрі для малоресурсного шифру «Кипарис» та показано, що структура цього шифру запобігає побудови ефективної атаки на основі цього методу.

Практичне значення результатів полягає в тому, що одержані результати можуть бути застосовними в побудові практичних атак на різні малоресурсні шифри, зокрема зі структурою ARX, а також для обгрунтування стійкості таких шифрів до диференціальних атак.

1 АКТУАЛЬНИЙ СТАН ТА ПЕРСПЕКТИВИ ДИФЕРЕНЦІЦІАЛЬНОГО АНАЛІЗУ ТА ARX-КРИПТОСИСТЕМ

У цьому розділі описується наявний метод диференціального криптоаналіза та його застосування до таких блокових шифрів як $SPECK$ та $CHAM$. Розглядаються класичні та сучасні методи генеруваня дифренціальних характеристик. Класичні методи включають пошук ймовірних диференціалів шляхом аналізу властивостей компонентів шифру. Сучасні методи, з іншого боку, використовують автоматизовані інструменти та алгоритми для ефективного пошуку характеристик з високою імовірністю. Також описані вище згаданих шифрів включно з описом малоресурсного шифру «Кипарис» детально описуються його структура, принципи роботи та застосування.

1.1 Диференціальний аналіз

Диференціальний криптоаналіз є потужним інструментом для аналізу симетричних криптографічних алгоритмів був вперше запропований Біхам і Шамір у 1991 році [2]. Його мета полягає у виявленні залежностей між змінами різниць на вході шифру та відповідними змінами різниць на виході, що дозволяє знайти слабкі місця алгоритму і, зрештою, зламати його. В основному був застосовний до симетричних блокових та потокових шифрів, має багато різновидів та типів атак для різних криптосистем для наприклад $ARX-$криптосистем тощо. Сьогодні стійкість до диференціальних атак є один з базових властивостей, якою повинен володіти хороший шифр. В даному розділі спочатку буде розглянуто загальний принцип диференціального аналізу, пізініше криптоаналіз $ARX-$криптосистем.

В роботі[4] диференціальний криптоаналіз описано наступнинм чином, суть полягає в аналізі пари шифрувань $P_1 \to C_1$, $P_2 \to C_2$ шляхом вивчення поширення вхідної різниці $\Delta P = P_1 \oplus P_2$ до вихідної різниці $\Delta C = C_1 \oplus C_2$ через шифр, відомий як слід або диференціальна характеристика. Диференціальна характеристика складається з послідовності різниць:

$$\Delta P \to \delta_1 \to \delta_2 \to \delta_3 \to ... \to \delta_{r-1} \to \Delta C.$$

Для здійснення атаки аналітику потрібена диференціальна характеристика із достатньо високою ймовірністю:

$$p = Pr_P[\Delta P \to ... \to \Delta C]$$

яка визначається як ймовірність для всіх відкритих текстів. Однак, для простоти аналізу та через наявність раундових ключів у шифрах, він зазвичай апроксимується ймовірністю характеристики над передбачуваними незалежними раундовими ключами. У цьому випадку ймовірність характеристику можна обчислити просто як добуток ймовірностей усіх окремих переходів. Для здійснення атаки аналітику потрібен диференціальна характристика із достатньо високою диференціальною ймовірністю:

$$p = Pr_P[\Delta P \to \delta_1] \cdot Pr_P[\delta_1 \to \delta_2] \cdot ... \cdot Pr_P[\delta_{r-1} \to \Delta C]$$

Кращу оцінку диференціальної ймовірності можна отримати, зібравши всі характеристики, які мають однакові вхідні та вихідні відмінності:

$$p = Pr_P[\Delta P \to \Delta C] = \sum_{\delta_1 ... \delta_{r-1}} Pr_P[\Delta P \to \delta_1 \to \delta_2 \to ... \to \delta_{r-1} \to \Delta C].$$

Подібно до класичного, диференціальний аналіз ARX-криптосистем також розгялядає різниці вхідних та вихідних шифротекстів за операцією

$\oplus$, оскільки операції циклічного зсуву та побітового додавання є лііними відносно операції $\oplus$, то можливі диференціали проходять з ймовірністю 1. Однак, вся складність полягає у визначенні диференціалів та їх ймовірностей через проходження опереції модульного додавання.

Означення 1.1 ([6]). xdp^+ це ймовірность, з якою вхідні різниці XOR α, β поширюються на вихідну різницю *XOR* γ через операції *ADD* і *SUB* відповідно, обчислені для всіх n-бітових входів α, β:

$$xdp^+(\alpha, \beta \to \gamma) = Pr_{x,y}\{(x + y) \oplus ((x + \alpha) + (y + \beta)) = \gamma\}$$

В роботі [6] ввели означення xdp^+ 1.1 описалии довели ряд важливих властивостей, в тому числі статичстичні. Позначимо $\omega = (\alpha, \beta \to \gamma)$ як довільний диференціал, тоді $xdp^+(\omega)$ випадкова функція розподіл, який індукується n-бітовими незалежними та рівноімовірними випадковими верторами α, β, γ.

Лема 1.1 ([6]). *xdp є інваріантивними відносно будь якої перестановки вхідних даних:*

$$xdp^+(\alpha, \beta \to \gamma) = xdp^+(\beta, \alpha \to \gamma) = xdp^+(\gamma, \beta \to \alpha) = ...$$

Лема 1.2 ([6]). $P(xdp^+(\omega) \neq 0) = \frac{1}{2}\left(\frac{7}{8}\right)^{n-1}$. *Нехай* xdp $0 < k < n$, *тоді* $P(xdp^+(\omega) = 2^{-k}) = \frac{1}{2}\left(\frac{7}{8}\right)^{n-1} B(k, n-1, \frac{6}{7})$.

Лема 1.3 ([4]). *Нехай α, β незалежні та рівноімовірні випадкові n-бітові вектори, тоді очікувана кількість диференціалів γ така, що перехід $(\alpha, \beta) \to \gamma$ є можливим, тобто $xdp^+((\alpha, \beta \to \gamma) > 0$, наступна:*

$$\underset{\alpha,\beta}{E}[|\{\gamma : xdp^+(\alpha, \beta, \gamma) > 0\}|] = \left(\frac{7}{4}\right)^{n-1} = 2^{(n-1)\log_2 \frac{4}{7}}.$$

Лема 1.4 ([4]). *Нехай α, β незалежні та рівноімовірні випадкові n-бітові вектори та $(\alpha, \beta) \to \gamma$ буде переходом через операцію модульного додавання, серед усіх можливих переходів переході, тоді*

середня диференціальна ймовірність переходу p визначається як:

$$\underset{\substack{\alpha,\beta,\gamma \\ xdp^+(\alpha,\beta,\gamma)>0}}{E}[xdp^+(\alpha,\beta,\gamma)] = \left(\frac{4}{7}\right)^{n-1} = 2^{(n-1)log_2\frac{4}{7}}.$$

В даній роботі будується фільтр для можифікованого шифру «Кипарис-256», тож доцільно оцінити очікувану кількість та середню ймовірність можливих диференціалів для нього. Очікувана кількість для 32-бітного модульного додавання $2^{25.028}$ та відповідно середня ймовірність $2^{-25.028}$, а середня вага диференціала 26.57 Операція модульного додавання у модифіковоного шифру використовується послідовно 4 рази. Отже оцінка зверху очікуваної кількості диференціалів $2^100.112$ та відповідно ймовірність $2^-100.112$

1.2 Алгоритми пошуку диференціалів та їх ймовірностей

Перед початку опису основних алгоритмів необхідно ввести декілька додаткових означень та алгоритмів.

Нехай $x, y, z \in \Sigma^n$ n-бітові слова, x_i, y_i, z_i відповідний i-й біт слова, де $i \in \{0, n-1\}$, тоді $eq(x,y,z) = (\neg x \oplus y) \wedge (\neg \oplus z)$, іншими словами $eq(x,y,z)_i = 1 \iff x_i = y_i = z_i$, $xor(x,y,z) = x \oplus y \oplus z$.

Необхідно ввести поняття *aop* (all-one parity) це функція яка на вході приймає n-бітне слово та повертає n-біне слово $y = aop(x)$, де $y_i = 1$ коли найдовша послідовність $x_i x_{i+1} x_{x+2} ... = 111..1$ має непарну довжину. Також важлива функція $c(x,y)$ (common alternation parity) яка приймає на вхід два n-бітових слова, де

$$C(x,y) = aop(\neg(x \oplus y) \wedge (\neg(x \oplus y) \gg 1) \wedge (x \oplus (x \gg 1))).$$

Також для функцій $aop(x)$ та $C(x,y)$ також існують їхні дуальні відповідники з оберненими аргументами на вході, $aop^r(x) = aop(x^{'})$ та $C^r(x^{'}, y^{'})$, $x^{'}_n = x_{n-i}$ і $y^{'}_n = y_{n-i}$.

Algorithm 1.1 Алгоритм для обчислення $aop(x)$

Require: $x \in \Sigma^n$, n - степінь 2
Ensure: $aop(x)$
  $x[1] = x \wedge (x \gg 1)$;
  for $i \leftarrow 2$ to $\log_2 n$ **do**
    $x[i] \leftarrow x[i-1] \wedge (x[i-1] \gg 2^{i-1})$;
  end for
  $y[1] \leftarrow x \wedge \neg x[1]$;
  for $i \leftarrow 2$ to $\log_2 n$ **do**
    $y[i] \leftarrow y[i-1] \vee ((y[i-1] \gg 2^{i-1}) \wedge x[i-1])$;
  end for
  return $y[\log_2 n]$;

Рисунок 1.1 – Алгоритм для обчислення $aop(x)$

Для подальшої роботи необхідно вміти обчислювати ймовірність переходу, тобто потрібно знати алгоритм знаходження xdp^+. В роботі Х. Ліпма та Ш. Моріай [6] було доведно низку тверженнь, теорем та розроблено алготимів для безпосереднього обчислення ймовірності проходежня диференціальної характеристики.

Теорема 1.1. *Нехай* $\delta = (\alpha, \beta \mapsto \gamma)$ *довільний диференціал. Алгоритм 1.2 повертає ймовірність* $xdp^+(\delta)$ *за* $\Theta(\log(n))$, *якщо бути більш точним, то обчислює ймовірність за* $\Theta(1) + t$, *де* t *це час обчсилення ваги диференціала* w_h.

Algorithm 1.2 Log-time algorithm for DP^+

1: **Вхід:** $\delta = (\alpha, \beta \mapsto \gamma)$
2: **Вихід:** $\mathrm{xdp}^+(\delta)$
3: **if** $\mathrm{eq}(\alpha \ll 1, \beta \ll 1, \gamma \ll 1) \wedge (\mathrm{xor}(\alpha, \beta, \gamma) \oplus (\beta \ll 1)) \neq 0$ **then**
4: **return** 0
5: **end if**
6: **return** $2^{-w_h(-\mathrm{eq}(\alpha \ll 1, \beta \ll 1, \gamma \ll 1))}$

Рисунок 1.2 – Алгоритм для обчислення ймовірності xdp^+

Для побудови атаки необхідно шукати всі можливі диференціали з певною ймовірністю тобто ті в яких ймовірність переходу вища за пеdний

поріг p. В роботі Х. Ліпма та Ш. Моріай [6] також наведено алгоритми пошуку всіх (α, β) оптимальних γ та алгоритм пошуку одного найбільш ймовірного. Алгоритм отримує на всіх два диференціали (α, β) та знаходить всі різниці γ, для яких $xdp^+(\alpha, \beta \to \gamma) = max_\gamma xdp^+(\alpha, \beta \to \gamma)$. Далі такі називатимуться γ оптимальні. При цьому коли оптимальний γ відомий то можна найти його ймовірнсть безпоседньо через алгоритм 1.2.

Algorithm 1.3 Алгоритм пошуку всіх $xdp^+(\alpha, \beta \to \gamma) = xdp^+_{max}(\alpha, \beta)$

Вхід: (α, β)
Вихід: всі оптимальні (α, β) диференціали γ
$\gamma_0 \leftarrow \alpha_0 \oplus \beta_0$
$\rho \leftarrow C(\alpha, \beta)$
for $i = 1$ to $n - 1$ **do**
  if $\alpha_i = \beta_i - 1 = \gamma_i - 1$ **then**
    $\gamma_i \leftarrow \alpha_i$
  else if $i = n - 1$ or $\alpha_i \neq \beta_i$ or $\rho_i = 1$ **then**
    $\gamma_i \leftarrow \{0,1\}$
  else
    $\gamma_i \leftarrow \alpha_i$
  end if
end for
return γ

Рисунок 1.3 – Алгоритм пошуку всіх $xdp^+(\alpha, \beta \to \gamma) = xdp^+_{max}(\alpha, \beta)$

Алгоритм повертає всі диффренціали з максимальними ймовірностіями, тож не потрібно їх додатково перевіряти та фільтрувати. Також Х. Ліпма та Ш. Моріай [6] представили Алгоритм 1.4 для пошуку одного оптимального дифренціала який працює за логарифмічнйи час. Хоча він і не підходить для активного використання та може бути використаним для пошуку одного оптимального, тобто найкращої диференціальної характеристики для попереднього аналізу та оцінок.

Звичайний метод гілок на границь потребує значного процесорного часу та об'му пам'яті, бо фактично обходить всі можливі варіанти, що по суті є дуже неоптмальним способом. А коли розмір слова n стає великим, то це стає практично неможливою задачею, навіть при застовуванні до одного раунда.

Algorithm 1.4 Алгоритм пошуку єдиного оптимального (α, β) γ диференціала

Вхід: (α, β)
Вихід: оптимальний (α, β) диференціал γ
$r \leftarrow \alpha \wedge 1$
$e \leftarrow \neg(\alpha \oplus \beta) \wedge \neg(r)$
$a \leftarrow e \wedge (e \ll 1) \wedge (\alpha \oplus (\alpha \ll 1))$
$p \leftarrow aop^r(a)$
$a \leftarrow (a \vee (a \gg 1)) \wedge \neg(r)$
$b \leftarrow (a \vee e) \ll 1$
$\gamma \leftarrow ((\alpha \oplus p) \wedge a) \vee ((\alpha \oplus \beta \oplus (\alpha \ll 1)) \wedge \neg(a) \wedge b) \vee (\alpha \wedge \neg(a) \wedge \neg(b))$
$\gamma \leftarrow (\gamma \wedge \neg(1)) \vee ((\alpha \oplus \beta) \wedge 1)$
return γ

Рисунок 1.4 – Алгоритм пошуку одного оптимального (α, β) γ диференціала

Саме тому в роботі [5] Huang і Wang запропонували покращений метод пошуку диференціальних характеристик для ARX-криптосистем, який вирішує цю проблему. Вони зменшили складність пошуку видаливши неможливі кортежі (α, β, γ) модульного додавання на ранніх етапах пошуку. Алгоритм фактично є прокращеним варіантом Алгоритму 1.3, тільки на відміну від нього працює точніше, гнучкіше та ефективніше через можливість вибору порогу.

В своїй роботі [5] Huang і Wang імпееметували та застосували свій пошук для блокових ARX шифрів $SPECK$, $SPARX$ та $CHAM$. Якщо розгядяати структуру метода, то він складається з декількох основних кроків і може будти застосовним до інших ARX подібних шифрів. Алгоритм пошуку може бути застосовним до будь якого блокового шифру, проте потребує модифікації для різних блокових шифрів, проте також має спільні риси. Якщо описувати коротко, то алгоритм має 4 етапи. Етап перший це процес переобчислення та зберігання таблиць. В другій відбувається генерація правильних кортежів $(\alpha_{i,j}, \beta_{i,j}, \gamma_{i,j})$, загальною процедурою, поступово збільшуючи ваги на перших r_1 раундах. Третій етап подібний до другого тільки процес проводиться на середній раундах розділючи вхідні $(\alpha_{r_m,j}, \beta_{r_m,j}))$ на n/m m-бітові

підблоки, також заускається інша відміна процедура від другуо етапу. Останній четвертий етап це ітерації третього етапу до останього раунда перевіряючи відповідну умову.

Існую декілька основних підходів до методів пошуку диференціальних характеристик, а саме використання еволюційних алгоритмів, які базуються на ідеях природнього відбору, селекцій, мутацій, схрещування тощо. Інший же підхід метод лінійного апроксимованого криптоаналізу, який базується на лінійних апроксимацій і використовує лінійні функції для апроксимації нелінійних функцій і в залежності від параметрів може знаходити диференціальні характеристики з високою ймовірністю. Зараз також набрає популяність використання нейроних мереж в криптоаналізі, для виявлення складних залежностей.

Підсумовуючи, сучасні методи є комбінацією теорії та експериментів в крипоаналізі, які алгоритми та автоматизовані інструменти для швидкого та ефективного знаходження диференціальних характеристик.

1.3 Опис блокового малорусурсного шифру «Кипарис»

Блоковий шифр «Кипарис» [8] розроблений для малоресурсних пристроїв. В даній роботі будується диференціальна атака на її модифіковану версію, нижче описується детальний необхідний опис шифру.

Блоковий шифр «Кипарис» підтримує дві конфігруації для 32-бітної та 64 бітної архітектури відповідно «Кипарис-256» та «Кипарис-512». Алгоритм шифрування «Кипарис» виконується над блоками розміром l і з використанням ключа довжиною k, де l,$k \in \{256, 512\}$, $l = k$. Операції виконуються над s-блоками розміром, де $s \in \{32, 64\}$ відповідно. Повна інформація про конфігурації наведені в таблиці 1.1. Всі подільші атаки в даній роботі будуть зосередження на модифікованій версії «Кипарис-256». Модифікація полягає в тому що урізається шифрується перетворення і

Таблиця 1.1 – Параметри блокового шифру «Кипарис»

	Кипарис-256	Кипарис-512
Розмір блока (l), бітів	256	512
Довжина ключа (k), бітів	256	512
Довжина слова (s), бітів	32	64
Кількість ітерацій перетворення (t), бітів	10	14
(r_1, r_2, r_3, r_4)	(16, 12, 8, 7)	(23, 24, 16, 15)

замість двух послідовний h застосовується лише одна функція.

На вході шифруючого перетворення «Кипарис» подається відкрийти текст $P = (P_0,\ P_1,\ P_2,\ ...,\ P_7)$, $K = (K_0,\ K_1,\ ...,\ K_7)$ - секретний ключ, раундові ключі $RK^{(0)}$, $RK^{(1)}$, ..., $RK^{(t-1)}$, де кожен ключ $RK^{(i)} = (RK_0^{(i)}$, $RK_1^{(i)}$, $RK_2^{(i)}$, $RK_3^{(i)})$ це 4 s-бітові слова.
Спочатку P розбивається на $L = (P_0,\ P_1,\ P_2,\ P_3)$ і $R = (P_4,\ P_5,\ P_6,\ P_7)$
Результат кожного раунда раунда:

$$\begin{aligned} L_i &= R_{i-1} \oplus F(L_{i-1}, RK^{(i-1)}) \\ R_i &= L_{i-1} \end{aligned} \tag{1.1}$$

де $F(L_{i-1}, RK^{(i-1)}) = h(h(L_{i-1} \oplus RK^{(i-1)}))$
$h(\alpha) = h(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \to (\alpha_0^{'}, \alpha_1^{'}, \alpha_2^{'}, \alpha_3^{'}) = \alpha_0^{'} \parallel \alpha_1^{'} \parallel \alpha_2^{'} \parallel \alpha_3^{'} = \alpha^{'}$

1. $\alpha_0 = \mathrm{ADD}(\alpha_0, \alpha_1)$
2. $\alpha_3 = \mathrm{XOR}(\alpha_0, \alpha_3)$
3. $\alpha_3 = \alpha_3 \lll r_1$
4. $\alpha_2 = \mathrm{ADD}(\alpha_2, \alpha_3)$
5. $\alpha_1 = \mathrm{XOR}(\alpha_1, \alpha_2)$
6. $\alpha_1 = \alpha_1 \lll r_2$
7. $\alpha_0 = \mathrm{ADD}(\alpha_0, \alpha_1)$
8. $\alpha_3 = \mathrm{XOR}(\alpha_0, \alpha_3)$
9. $\alpha_3 = \alpha_3 \lll r_3$
10. $\alpha_2 = \mathrm{ADD}(\alpha_2, \alpha_3)$
11. $\alpha_1 = \mathrm{XOR}(\alpha_1, \alpha_2)$
12. $\alpha_1 = \alpha_1 \lll r_3$

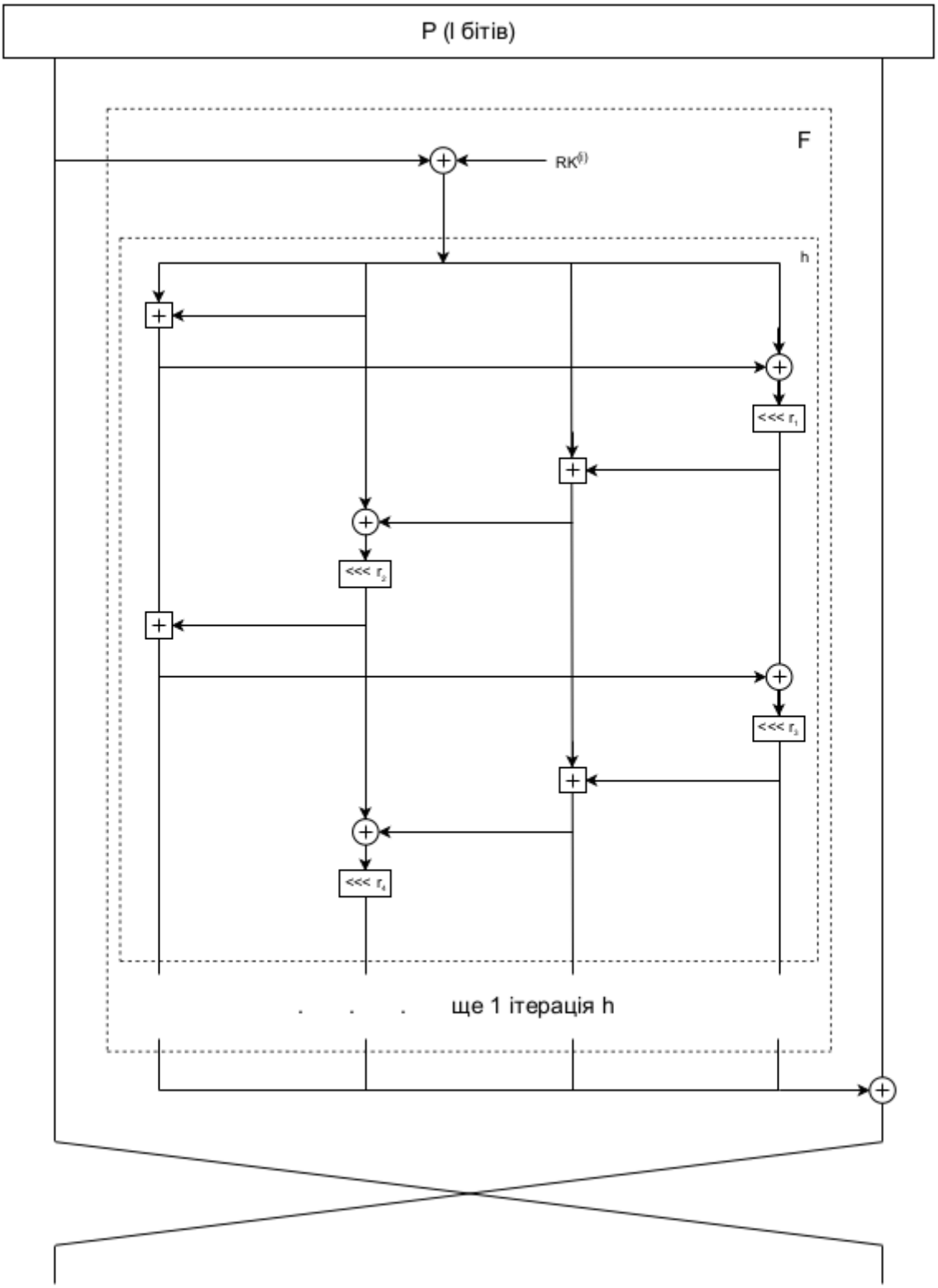


Рисунок 1.5 – Раундова функція оригінального шифру «Кипарис».

1.4 Опис блокового шифру $SPECK$

Speck сімейство малоресурсних блокових шифрів розроблений Агентством національної безпеки США, оптимізований під програмну реалізацію, основню цілю є реалізація для мікроконтролерів. Speck заснована на конструкції $add - rotate - xor$ (ARX) структуру, пітримує розмір блоку 32, 48, 64, 128. Далі описано короктий опис шифруючого перетворення та схеми розгортатання колючів, детільніше про $SPECK$ [1].

Оскільки $SPECK$ Фейстелю подіний шифр, то і шифруюче перетворнення у нього відповідне. Розмір блоку $SPECK2n$, складається з двух $n-$бітих слів та ключем розміром kn, де k це кількість слів. Ранудова функція Speck2n $R_K : \{0,1\}^{2n} \to \{0,1\}^{2n}$

$$R_K(x,y) = ((x \ggg r_a) + y) \oplus K, (y \lll r_b) \oplus (((x \ggg r_a) + y) \oplus K),$$

де $r_a = 7$, $r_a = 2$ при $n = 16$ та $r_a = 8$, $r_a = 3$ для інших n.

Схема розгортання ключів $SPECK$ є вже досить добре вивченою , проте важливо розуміти її структору. На вхід схема розгортання ключів бере на вході головний ключ та генерує R раундових ключів $K_0, K_1, ..., K_{R-1}$, де R кількість раундів. При шифрувані кожен раундовий ключ використовується для відповіного раунда. Для розшифрування рундові ключі подаються у зворотньому порядку.

Нехай K ключ для $Speck2m$. Записуємо $K = (l_{m-2}, ..., l_0, k_0)$, де $l_i, k_0 \in GF(2)^n$ для значенн $m \in \{2,3,4\}$. Значення k_i це раундовий ключ для і-го раунд, послідновності k_i та l_i задаються:

$$l_{i+m-1} = (k_i + (l_i \ggg r_a)) \oplus i,$$

$$k_{i+1} = (k_i \lll r_b) \oplus l_{i+m-1}.$$

1.5 Опис блокового шифру $CHAM$

Блоковий шифр $CHAM$ вперше був представлений в ICISC 2017 [7], розроблений як малоресурсний, чотирьох гілкових оснований на загальній схемі Фейстеля, який комібнує хороші рішення з блокових шифрів $SIMON$ та $SPECK$. $CHAM - n - k$ також преставлений як сімейство блокових шифрів оскільки має низку різних конфігурацій під різні потужності та системи $CHAM - 64/128$, $CHAM - 128/128$, $CHAM - 128/256$ з 88, 112 та 120 раудами відповідно, де n довжина блока і k як довжина секретного ключа. Більш детально про конфігурації в таблиці 1.2.

Таблиця 1.2 – Параметри блокового шифру $CHAM$

	$CHAM$-64/128	$CHAM$-128/128	$CHAM$-128/256
Розмір блока (n), бітів	64	128	128
Довжина ключа (k), бітів	128	128	256
Довжина слова (s), бітів	16	32	32
Кількість ітерацій перетворення (t)	88	112	120

На рисунку 1.6 показано два послідовних раунди. У статті [3], де розглядається побудова атаки, автори сформулювали декілька тверджень на основі їхніх спостережень, а саме про кластеризацію 1.1 та 1.2. Пропустимо детальний опис схеми розгортання ключів, оскільки в даному випадку він не потрібний.

Твердження 1.1. *У будь-яких 4 раундах $CHAM$ різниця між вхідними та вихідними даними разом визначає повний 4-раундовий диференціальний шлях. І тому 4 раунди $CHAM$ не мають жодного кластерування шляхів.*

Твердження 1.2. *Додавання в i-му раунді $CHAM$ використовує як вхідні дані результати додавання в $(i-3)$-му та $(i-4)$-му раундах. Віднімання (при розшифруванні) в i-му раунді $CHAM$ використовує як*

вхідні дані результати віднімання в $(i+1)$*-му та* $(i+4)$*-му раундах.*

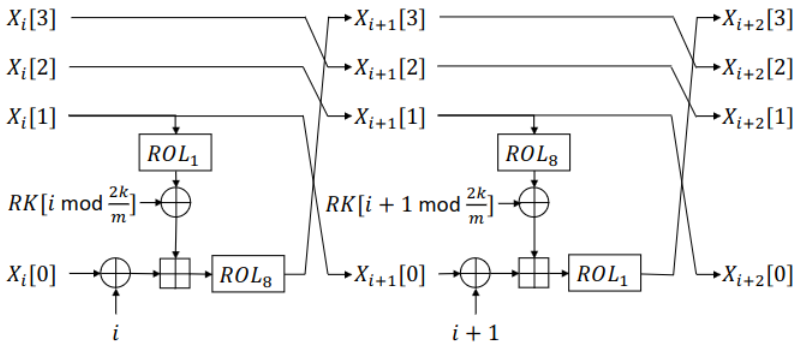


Рисунок 1.6 – Два послідновнх раунда $CHAM$

1.6 Загальний опис атаки Зустріч у Фільтрі

В даному розділі буде розглянуто техніку диференціального криптоаналізу Зустріч у Фільтрі ($Meet-in-the-filter$ або MiF)[4], метод складається з двух основних частин, а саме метод MiF для пошуку найкращих диференціальних характеристик і процедура відновлення ключа оснований на динамічному підрахунку. MiF застосовний до шифрів з відносно повільною дифузією. В даній роботі необхідно розуміти тільки на метод MiF, тож опис процедури відновдення ключа пропущена.

Нехай блоковий шифр з $r+u$ раундами із r-раундовою диференціальною характеристикою та u-раундових диференціальних характеристик з процедури звортонього пошуку для процедури відновлення ключа. В свою чергу u можна розділити на s і t для знаходження компромісу між часом та пам'яттю. Множини s і t шукаються незалежно одне від одного та в результаті відбувається пошук співпадінь в множинах (зустріч посередині). На рисунку 1.7 проілюстровна загальна схема фільтра.

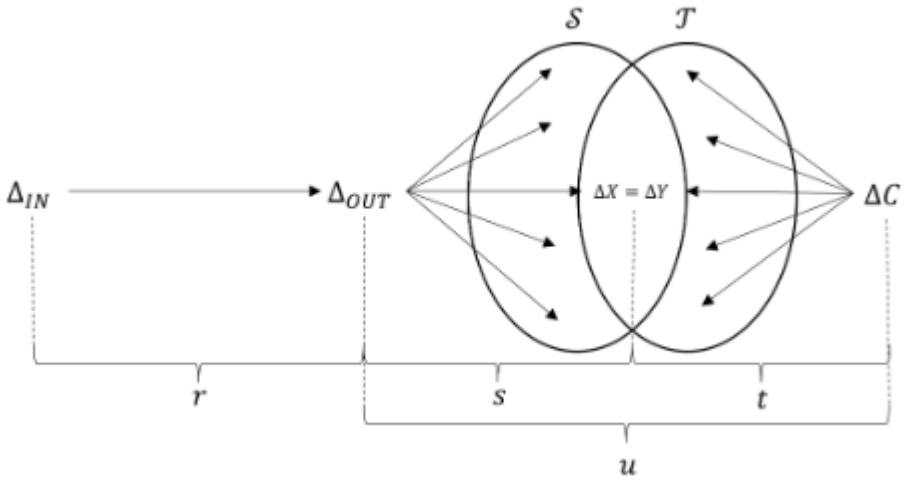


Рисунок 1.7 – Загальна структура фільтра Зустірч у Фільтрі

Метод починає роботу з $r-$раундового диференціалу з деякою ймовірністю p, позначеною як $\Delta_{IN} \xrightarrow{\mathrm{r}} \Delta_{OUT}$. Обирається відповідне розподілення u на s верхніх і t нижніх раундів ($u = s + t$) разом із відповідними ймовірностями 2^{-w_s} і 2^{-w_t}.

Означення 1.2. Кластер $\mathcal{S}(s, w_s)$ є множиною всіх s-раундових диференціальних характеристик τ_s починаючи з різниці Δ_{OUT} та має ймовірність не менше за 2^{-w_s}:

$$\mathcal{S}(s, w_s) = \left\{ \tau_s = \left(\Delta_{OUT} \xrightarrow{\mathrm{S}} \Delta_X \right) : Pr[\tau_s] \geqslant 2^{-w_s} \right\} \tag{1.2}$$

Позначення $\mathcal{S}(s, w_s)$ говорить про те, що множина $\mathcal{S}$ є функцією від параметрів s і w_s. Надалі буде використовуватись $\mathcal{S}$ як скорочення для $\mathcal{S}(s, w_s)$, коли параметри очевидні з контексту. Побудова $\mathcal{S}$ зазвичай вимагає незначного часу попереднього обчислення порівняно з повними диференціальними атаками.

Означення 1.3. Фільтруюча множина τ складається з усіх t-раундових характеристик τ_t, починаючи з ΔC у зворотному напрямку і мають ймовірність не менше 2^{w_t}.

$$\mathcal{T}(t, w_t) = \left\{ \tau_t = (\Delta C \xrightarrow{\mathrm{t}} \Delta Y) : Pr[\tau_t] \geqslant 2^{-w_t} \right\}. \tag{1.3}$$

Подібно до $\mathcal{S}(s, w_s)$, множина $\mathcal{T}(t, w_t)$ є функцією від t і w_t. Вподальшому $\mathcal{T}(t, w_t)$ позначатиметься як $\mathcal{T}$, якщо параметри очевидні з контексту. Всі наші τ_t в $\mathcal{T}$, залишаємо тільки ті, в яких вихідна різниця ΔY збігається з різницею вихідного ΔX з τ_s в $\mathcal{S}$. Співпадіння між отриманими $\tau_s \in \mathcal{S}$ та $\tau_t \in \mathcal{T}$, в$u-$раунді характеристиці τ_u можна отримати наступною конкатенацією:

$$\tau_u = (\tau_t || \tau_s) = \Delta C \xrightarrow{\text{u}} \Delta_{OUT} = \left(\Delta C \xrightarrow{\text{t}} (\Delta Y = \Delta X) \xrightarrow{\text{s}} \Delta_{OUT}\right) \qquad (1.4)$$

Пара шифротекстів в яких відбувається співпадіння, далі запам'ятовується як кандидат *хороша пара*, тобто пара для відповідних відкритих текстів (P_1, P_2) який має пройдений диференціальну характеристику $(\Delta_{IN} \xrightarrow{\text{t}} \Delta_{OUT})$. До кожної такої пари додається набір запропонованих u-образних дифренціальних характеристик $\{\tau_u = \Delta_{OUT} \xrightarrow{s+t} \Delta C\}$. Останній містить інформацію для фази відновлення ключа і передається процедурі відновлення ключа. Множина $\mathcal{S}$ називається кластером, а процес зіставлення множини $\mathcal{T}$ з $\mathcal{S}$ називається (зворотним) фільтром. Абсолютні значення порогових значень ймовірностей за логарифмом основи 2, тобто константи w^s та w^t, називаються відповідно *вагою кластера* та *вагою фільтра*. Оскільки розбиття $s + t$ можна розглядати як один великий u-подібний фільтр, який пропускає лише праві пари-кандидати, процедура називається *"зустріч у фільтрі"* або MiF. В цілому MiF пропонує зловмиснику зменшення складності фільтрації для нижніх раундів u через компроміс між часом і пам'яттю.

Важливим аспектом складності атики Зустріч у Фільтрі це залежність від обраного диференціала, а особливо від вхідної різниці Δ_{OUT}, розбиття раундів, ваг кластерів $\mathcal{S}$, $\mathcal{T}$ та фільтрів w_s, w_t. Це напряму впливає на властивості диференціальних характеристик та безпосередньо на складність. Теоретично оцінiти розподіл ваг фактично неможливо через його непередбачуваність подальшою глибиною

диференціала. Також практично очевидно, що часова складінсть досить чутлива до ймовірністей проміжних диференціалів.

1.7 Застосування атаки Зустріч у Фільтрі до шифру $SPECK$

В даному розділі коротко описано короткі відомості результатів атаки Зустріч у Фільтрі на блоковий шифри $SPECK$. Результати атаки описано в роботі [4], було проведено атаки на різні раунди з відповіндними різними конфігураціями. описано як фільрація диференціальних характеристик так і відновлення ключа за допомгою динамічного підрахунку, оскільки в дана робота орінтована на пошук та фільтрування диференціальних характеристик, то опис процедури відновлення ключа опустим. Далі описано опис фільтрації до $SPECK32$ та оцінка складності.

Опис фільтра починається наступним чином, розглядаємо $SPECK$ як блоковий шифр з $r + u$ ранудами. Для відновдення повного ключа необхідно лише 4 підключа нижніх раундів, з яких можна дістати головний ключ. Отже, потрібно просто застосувати процедуру фільтрація з конфігурацією $(s, t) = (2,\ 2)$, що проілюстровано на 1.8. Елементи, виділені зеленим кольором, є фіксованими значеннями від найкращого диференціала r-раунду (trail), що використовується в атаці. Темно-жовті елементи походять із попередньо обчислених кластерних trails $\tau_s \in \mathcal{S}$. Фіолетовий елементи відповідають слідам $\tau_t \in \mathcal{T}$, створеним процедурою зворотного пошуку (зворотний фільтр).

$SPECK$ є шифром оснований на основі схеми Фейстеля, тому збіг елементів між множинами $\mathcal{S}$ і $\mathcal{T}$ може бути виконано ефективно по n біт за раз. Перша $n-$бітова перевірка виконується на правій гілці раунду $r + 3$ внизу 1.8. Вона порівнює різниці, згенеровані операцією ADD в останньому раунді, з різницями у правій гілці, що надходять з кластера $\mathcal{S}$. Ця відповідність ілюструється червоною лінією на Рисунку 1.8 (позначена як "First n-bit check"). Тільки шляхи $\tau_t \in \mathcal{T}$, які пройшли

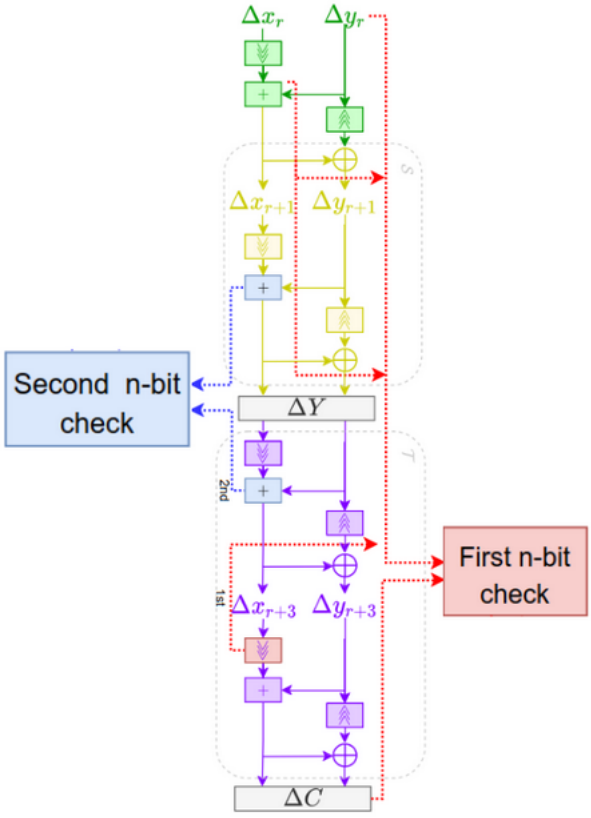


Рисунок 1.8 – Робота фільтра $2+2$ MiF на чотирьох нижніх раундах $r+4$ раундів атаки на Speck2n.

першу перевірку, переходять до другої n-бітової перевірки. Остання виконується на лівій гілці на раунді $r+2$ і показана синьою лінією на рисунку 3 (позначена як "Second n-bit check").

1.8 Застосування атаки Зустріч у Фільтрі до шифру $CHAM$

В даному розділі більш детальніше опишеться застосування MiF до $CHAM$, проте через специфіку атаки спочатку небхідно коротко розказати про те як шукали диференціальні характеристики в [5], потім результати та оцінки застосування MiF до різних кофігурацій $CHAM$.

Вхідну різницю $\Delta X_r = X_r \oplus X_r^{'}$ r раунда можна розписати як $\Delta X_r = \Delta X_r[0] \parallel \Delta X_r[1] \parallel \Delta X_r[2] \parallel \Delta X_r[3]$, де $\Delta X_r[j] \in \mathbb{F}_2^w$, $0 < \mathrm{j} < 3$.

Твердження 1.3 ([5]). *Нехай* ΔX_r, ΔX_{r+1} *вхідна та вихідна*

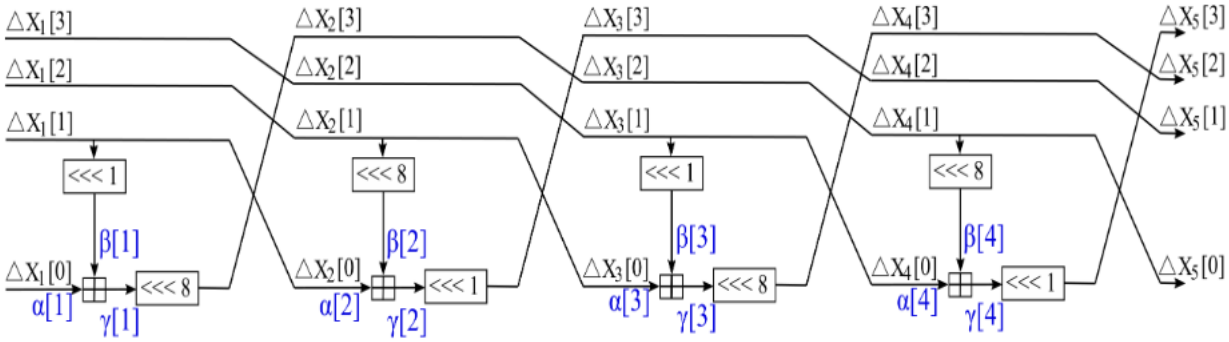


Рисунок 1.9 – Розповсюдженя різниць в чотирьох раундах $CHAM$

різниці r-ого раунда шифру $CHAM$*, тоді* $\Delta X_{r+1}[0] = \Delta X_r[1]$, $\Delta X_{r+1}[1] = \Delta X_r[2]$, $\Delta X_{r+1}[2] = \Delta X_r[3]$ *і* $\Delta X_{r+1}[3] = \delta_{Pr}(\Delta X_r[0], \Delta X_r[1] \lll r_a) \lll r_b$, *де* $\gamma = \delta_{Pr}(\alpha, \beta)$ *це вихідна різниця* γ *після подульного додавання згенеована із вхідних* (α, β) *з диференціальною ймовірністю* Pr.

Далі власне на основі 1.3 і будується алгоритм для генерування диференціалів, який дальше готовий до використання. Зосереджуватись на досліжджені цього алгоритму немає потреби, оскільки в даній роботі буде використовуватись інший метод.

На основі алгоритму в [3] побудувади атаку Зустріч у Фільтрі на 52-й раунд $CHAM - 64$ з кофігурацією (4+40+4+4). Із твердедження 1.1 можна зробити висновок, що конфігурації потрібно виключно кратно 4, бо при лише одному раунді шифрується четверта частина вхідного слова і відповідно так змінюється різниця. Вони використовували диференціал над 40 раундами з ймовірністю $p = 2^{-60.05}$. В сумі було сформовано $2^{95.72}$ пар ВТ-ШТ для анаізу, кожна з яких має кожна з яких є кандидадом після $4 + 40$ раундів. За попередніми розрахунками очікувалось найти як мінімум один хорошу пару в множині і кожна така пара за MiF отримує список кандидатів для останіх 8 раундів.

За результатами було знайдено, що кожна різниця шифротексту індукує $2^{64.84}$ диференціальних характеристик в середньому, що дає $2^{96.56}$ повних 52-х раундових дифференціальних характеристик з відповідними відкритими та шифротекстами, для подальшого аналізу. Далі робота

зосереджувалась на вгадуванні підключів в певному підібарному порядку з відпоіднимим перевіркам підключів.

Висновки до розділу 1

У даному розділі розглянуто основні аспекти диференціального аналізу, який є важливим методом криптоаналізу симетричних криптосистем та його застосування до ARX-криптосистем. Проаналізовано класичні роботи в цій галузі, що створюють основу для подальшого дослідження та розвитку методів диференціального криптоаналізу для ARX-криптосистем. Описані блокові шифри $SPECK$, $CHAM$ та Кипарис та проаналізовані їхні архітектурні особливості та ключові характеристики. А також представлені та проаналізовано класичні алгоритми для пошуку оптимальних дифренціалів, що будуть актично використовуватись та досліджуватись в подальшій роботі.

Також розглянути відносно нову техніку диференціального криптоаналізу Зустір у Фільтрі в контексті побудови фільтра і відповідно проаналізово застосовність цієї атаки наприкладі блокових шифрів $SPECK$ та $CHAM$. Розглянуто структури та особливості відповідних побудованих фільтрів та методи пошуку диференціальних характеристик для фільтрів.

2 РОЗРОБКА ТА АНАЛІЗ АЛГОРИТМУ ПОШУКУ ДИФЕРЕНЦІАЛЬНИХ ХАРАКТЕРИСТИК ДЛЯ МОДИФІКОВАНОГО БЛОКОВОГО ШИФРУ «КИПАРИС»

В даному розділі будується алгоритм пошуку диференціальних характеритсик для модифікованого шифру «Кипарис», а також аналізується відповідний результат застосування. Також обґрунтовується помилковість роботи алгоритму1.3 пошуку всіх оптимальних диференцілів із аналізом похибки. І на кінець розробляється та обґрунтовується застосовінсть метода Зустірч у Фільтрі.

2.1 Пошук диференціальних характеристик

У даному розділі проводиться формується алгоритм пошуку диференціалів та відповідних ймовірностей для функції h з шифру «Кипарис-256». Спочатку розробиться покроковий алгоритм пошуку всіх кінцевих диференціалів, на основі цього алгоритму виведеться формула пошуку загальної ймовірності цього диференціала. Нарешті, у останьому розділі проводиться аналіз отриманих результатів з метою визначення найкращого диференціалу, його ймовірності і зрештою підхід до наступних диференціальних атак.

Спочатку необхдіно побудувати алгоритм пошуку всі дифренціальних характеристик. Розглянемо функцію h з раундової функції блокового шифру «Кипарис», мета полягає найти всі можливі виходи з функції h при відмому вході. Отже нехай на вході функції $h(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ тоді на виході очікуєтсья $(\gamma_1, \gamma_2, \gamma_3, \gamma_4)$. Тобто $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \rightarrow \{(\gamma_1, \gamma_2, \gamma_3, \gamma_4)\}$. З алгоритму 1.3 по пошуку всіх γ з $xdp^+(\alpha, \beta \rightarrow \gamma)$, Х. Ліпма та Ш. Моріай [6], будується алгоритм. З

вхідних $\alpha_1\alpha_2$ легко знаходимо множину усіх $\{\beta_1\}$

$$xdp^+(\alpha_1, \alpha_2 \to \beta_1). \tag{2.1}$$

Таким же чином знаходимо усі β_2, в результаті чого отримаємо множину усіх $\{(\beta_1, \beta_1)\}$

$$xdp^+(\alpha_3, ((\beta_1 \oplus \alpha_4) \lll r_1) \to \beta_2). \tag{2.2}$$

Продовжуємо першим на виході отримуємо γ_1 і доповнюємо множну послідовностей $\{(\gamma_1, \beta_1, \beta_1)\}$.

$$xdp^+(\beta_1, ((\beta_2 \oplus \alpha_2) \lll r_2) \to \gamma_1). \tag{2.3}$$

Останньою подібним чином знаходимо γ_3 і відповідно розширюємо множину послідновностей до $\{(\gamma_3, \gamma_1, \beta_1, \beta_1)\}$.

$$xdp^+((((\beta_2, ((\beta_1 \oplus \alpha_4) \lll r_1) \oplus \gamma_1) \lll r_3) \to \gamma_3) \tag{2.4}$$

В результаті знайдено проміжну множину послідновностей $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \to \{(\gamma_3, \gamma_1, \beta_2, \beta_1)\} \to \{(\gamma_1, \gamma_2, \gamma_3, \gamma_4)\}$ якої легко можемо далі можна обрахувати γ_2, γ_4 напряму через наступні формули (2.5) і (2.6):

$$\gamma_2 = (((\alpha_2 \oplus \beta_2) \lll r_2) \oplus \gamma_3) \lll r_4 \tag{2.5}$$

$$\gamma_4 = (((\beta_1 \oplus \alpha_4) \lll r_1) \oplus \gamma_1) \lll r_3 \tag{2.6}$$

Згодом рівняння (2.5) і (2.6) потрібні будуть потрібні для обчисляння ймовірності $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \to (\gamma_1, \gamma_2, \gamma_3, \gamma_4)$. Також, це фактично означає, що з вихідних даних можна обчислити β_1, β_2 безпосередньо і відповідно обчислити ймовірність напряму.

$$\beta_1 = (((\gamma_4 \ggg r_3) \oplus \gamma_1) \ggg r_1) \oplus \alpha_4 \tag{2.7}$$

$$\beta_2 = (((\gamma_2 \ggg r_4) \oplus \gamma_3) \ggg r_2) \oplus \alpha_2 \tag{2.8}$$

Формули (2.5) та (2.6) означають, що ці виходи мають більш пряму залежність між собою $\left(\gamma_3 \overset{\cdots}{\rightarrow} \gamma_2, \gamma_1 \overset{\cdots}{\rightarrow} \gamma_4\right)$, що продемонстровано в 2.1.

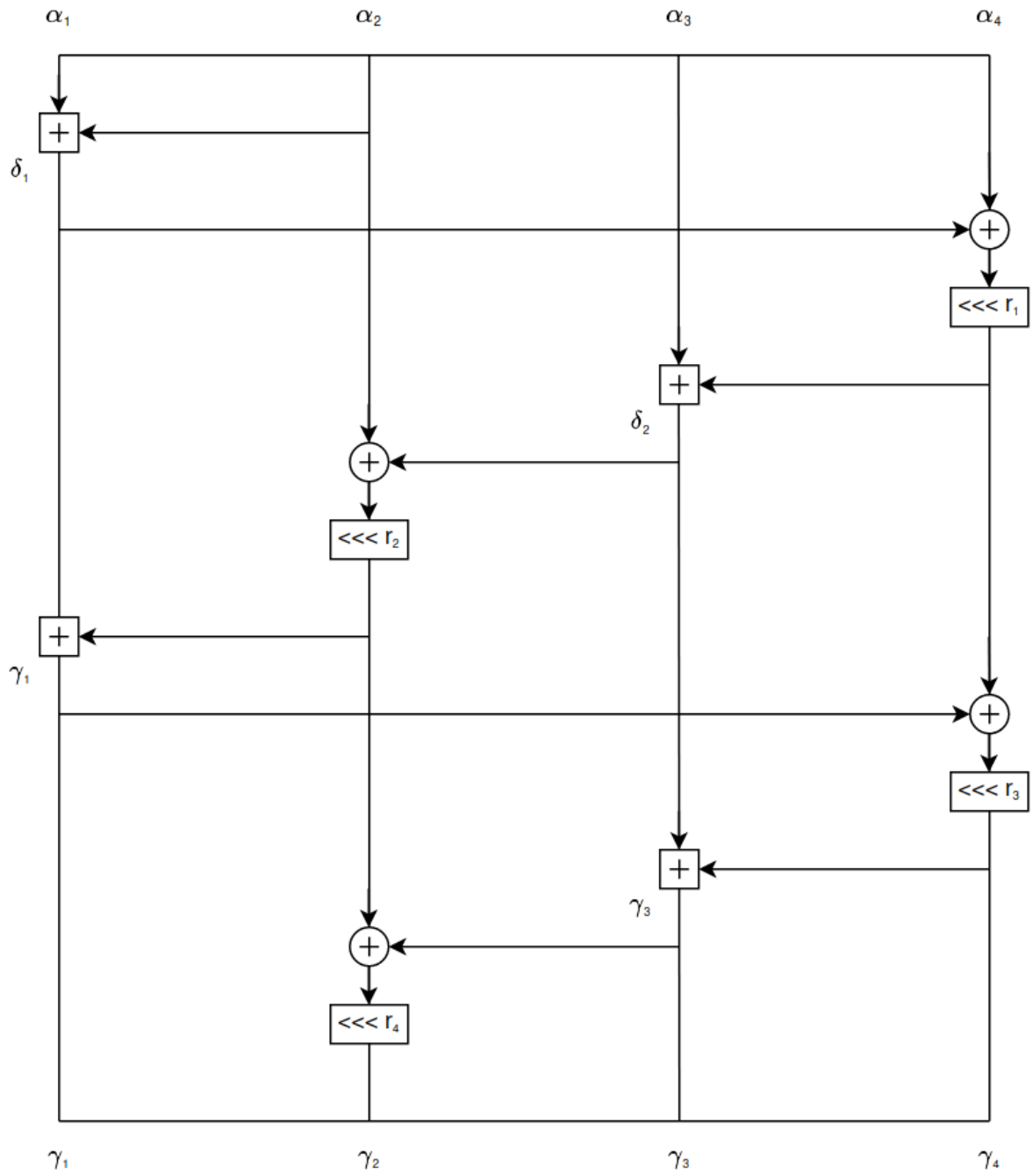


Рисунок 2.1 – Розташування проміжних диференціалів у функції h.

Щоб оцінити складність спочтаку потрібно обчислити ймовірність пошуку одного диференціала $P((\alpha_1, \alpha_2, \alpha_3, \alpha_4) \rightarrow (\gamma_1, \gamma_2, \gamma_3, \gamma_4))$. У функції h 4 модульного довання і з алгоритму пошуку множини

дифренціалів виплаває, що:

$$P\big((\alpha_1,\alpha_2,\alpha_3,\alpha_4)\to(\gamma_1,\gamma_2,\gamma_3,\gamma_4)\big)\approx xdp^+(\alpha_1,\alpha_2\to\beta_1)\cdot$$
$$\cdot\, xdp^+(\alpha_3,(\beta_1\oplus\alpha_4)\lll r_1\to\beta_2)\cdot$$
$$\cdot\, xdp^+(\beta_1,(\beta_2\oplus\alpha_2)\lll r_2\to\gamma_1)\cdot$$
$$\cdot\, xdp^+((\beta_2,(\beta_1\oplus\alpha_4)\lll r_1)\oplus\gamma_1)\lll r_3\to\gamma_3)$$

Згідно алгоритму[6] зможемо напряму обчислити ймовірності для всієї множини вихідних γ і отже відсіяти ті які мають занадто малу ймовіність p. Також потрібно розуміти, що деякі $\gamma_1,\gamma_2,\gamma_3,\gamma_4)$ можуть випливати з різних проміжних $(\gamma_3,\gamma_1,\beta_2,\beta_1)$ і тому їх ймовірність дещо вища. Якщо це враховувати, то результаті при фільтрації будуть точніші і можливо зменшить загальну кількість потрібних диференціалів через їх вищу ймовірність для більшої ефективності використання пам'яті та прискоренні процесу як такого.

При використані алгоритму 1.4 також легко найти найкращий з можливих дифереціалів і відповідно посто найти його ймовірність, однак при часто такий підхід може не призвести до результатів, бо в наступному раунді або як у функції h результат також залежить від циклічного зсуву та операції XOR і відповідно може бути відсутній подальший диференціал. Саме тому доцільно використовувати пошук усіх диференціалів за певним криетрієм і залишати лиш найбільш ймовірних. Проте в якості швидкої нескладної перервіки інколи доцільно таким практичним чином дізнатись найвищу ймовірність для ефективнішого відсіювання.

2.2 Аналіз результатів

В даному розділі проведено аналіз результатів у пошуку найкращого диференціалу та їх ймовірності. При аналізі були враховані закономірності вхідних параметрів. Результати цього дослідження дозволяють визначити

оптимальні стратегії для знаходження найефективніших атак на систему. Розглядаючи атаки такого типу, варто звернути увагу на числа з виключно старших бітів як потенційно корисні, оскільки вони часто перекривають одне одного при застосуванні операцій додавання за модулем n.

Необхідно проаналізувати вихідні γ та ймовірності при пошуку оптимальної диференціальної характеристики і відповіно далі описується застосування Алгоритму 1.4 до деяких вхідних різниць та аналізується на можливі вразливості функції h.

При атаках такого типу часто варто починати із чисел зі виключно страшних бітів, оскільки вони частіше можуть перекрити одне одного при додаванні за модулем n, також при побітовому циклічному зсуві вліво ($\lll$). Нехай $\delta_0 = \texttt{0x00000000}$, $\delta_1 = \texttt{0x80000000}$, $\delta_2 = \texttt{0xC0000000}$ 32-бітні диференціали, в таблиці 2.1 наведено пошук найкращого диференціала через функцію h з відпоідними ймовірностями.

Таблиця 2.1 – Приклад наймовірнішого диференціалу (старші біти)

Вхідний параметр α_i				Вхідний параметр γ_i				Ймовірність
α_1	α_2	α_3	α_4	γ_1	γ_2	γ_3	γ_4	P
δ_1	δ_0	δ_0	δ_0	0x88000000	0x802A002	0x2A002	0x88008000	0.015625
δ_0	δ_1	δ_0	δ_0	0x88000800	0x2802A802	0x2002A002	0x88008800	0.00390625
δ_0	δ_0	δ_1	δ_0	0x0000800	0xA0000800	0xA0000000	0x0000800	0.25
δ_0	δ_0	δ_0	δ_1	0x8000000	0x800A002	0x0000A002	0x8008000	0.03125
δ_2	δ_2	δ_0	δ_0	0x0000800	0x20000800	0x20000000	0x800	0.25
δ_2	δ_0	δ_2	δ_0	0x88000800	0xA802A802	0xA002A002	0x88008800	0.00390625
δ_2	δ_0	δ_0	δ_2	0x80000000	0x20000	0x20000	0x80000000	0.5
δ_0	δ_2	δ_2	δ_0	0x88000000	0x8802A002	0x8002A002	0x88008000	0.015625
δ_0	δ_2	δ_0	δ_2	0x80000800	0x20020800	0x20020000	0x80000800	0.125
δ_0	δ_0	δ_2	δ_2	0x8000800	0xA800A802	0xA000A002	0x8008800	0.0078125
δ_0	δ_0	δ_0	δ_0	0x	0x	0x	0x	1

На таблиці 2.1 показано, які з гілок є потенційно більш вразливі ніж інші. Одже, якщо розгладати гілки незалежно, тобто одна гілка має значення, всі решта нульові, то α_2 має найбільший вплив на всі інші

гілки, а найменший відповідно α_3. Цей висновок випливає з того, що якщо найвища ймовірність є низькою, то їх кілкьість повина бути значно порівняно якщо ймовірність є високою, значить кільскість дифференціальни характеристик не так багато. Далі більш наглядно розкривається ця тема.

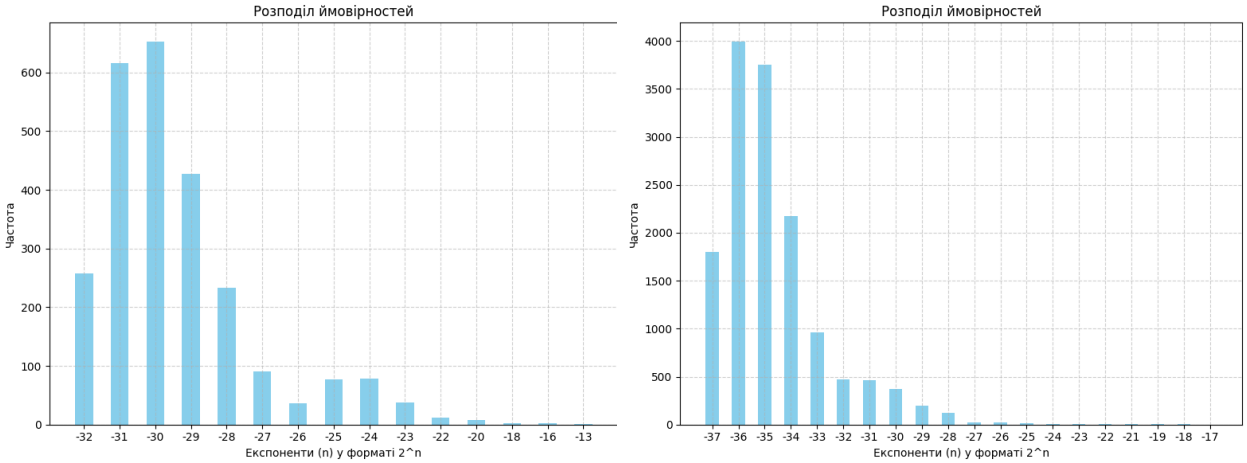
Тепер потрібно проаналізувати всі вихідні γ при пошуку всіх диференціальних характеристик. Нижче в розділі описано результати та спостереження щодо застосування Алгоритму 1.3 до декількох вибіркових різниць. Також були зроблені припущення щодо кількості можливих диференціалів через відповідну максимально можливу ймовірністю, відповідно проведено спроба навести декілька прикладів підтвердження даної гіпотези.

Таблиця 2.2 – Кількість диференціалів після застосування комінованого алгоритму

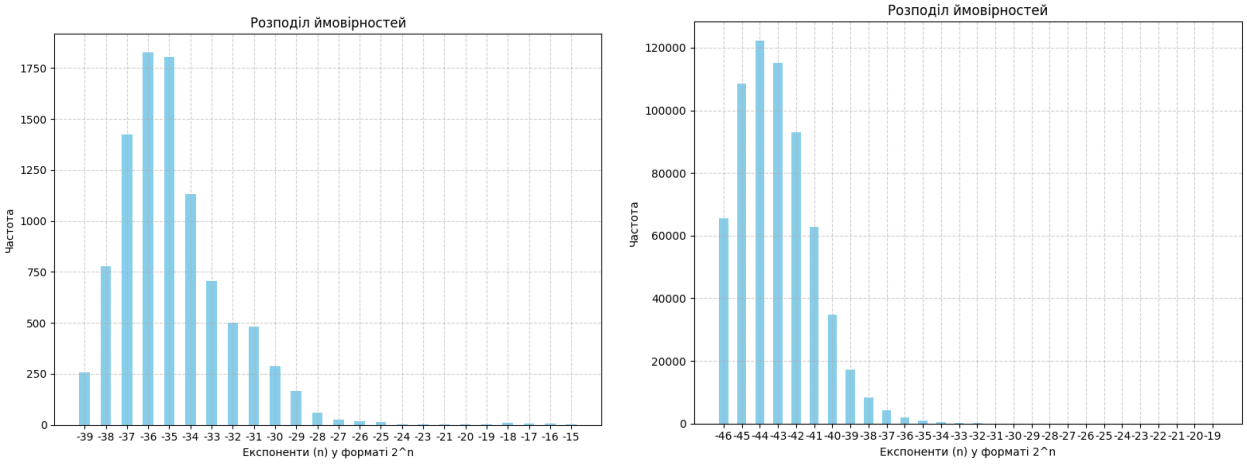
α_1	α_2	α_3	α_4	Кількість
0xF8000000	0x00000000	0x00000000	0x00000000	2531
0x00000000	0xF8000000	0x00000000	0x00000000	9514
0x00000000	0x00000000	0xF8000000	0x00000000	179
0x00000000	0x00000000	0x00000000	0xF8000000	1040
0xF8000000	0xF8000000	0x00000000	0x00000000	105437
0xF8000000	0x00000000	0x00000000	0xF8000000	98
0xFC000000	0x00000000	0x00000000	0x00000000	14382
0x00000000	0xFC000000	0x00000000	0x00000000	635875
0x00000000	0x00000000	0xFC000000	0x00000000	814
0x00000000	0x00000000	0x00000000	0xFC000000	7991
0xFC000000	0xFC000000	0x00000000	0x00000000	39445
0xFC000000	0x00000000	0x00000000	0xFC000000	301

За таблицею 2.2 можна помітити, що кількість диференціальних характеристик через функцію h за Алгоритмом 1.3 схожа за розподілом до кращих диференціалів, що певною мірою підтверджує припущення.

Проте одне з припущенн спрощується, а саме те що входи $(\delta_i, \delta_i, \delta_0, \delta_0)$ є оптимальними. Також варто звернути увагу на розподіл ймовірностей диференціальних характеристик на графіках 2.2, де чітко видно, що розподіл ймовірностей диференціальних характеристик схожий до нормального розподілу. Проте виникають питання щодо ефективності Алгоритму 1.3, оскільки кількість диференціалів занадто невелика для такого входу, що вказує на необхідність подальшого перегляду роботи цього алгоритму.



(а) 5 старших бітів на вході α_1 **(б)** 6 старших бітів на вході α_1



(в) 5 старших бітів на вході α_2 **(г)** 6 старших бітів на вході α_2

Рисунок 2.2 – Порівняння розподілів оптимальних диференціальних характеристик для різних входів на функцію h

За попередніми оціками кількість диференфіальних характеристик

значно менше ніж очікується. Насамперд такий результата може буди викликаний двома потенційними причинами, або алгоритм генерації диференціальних характеристик не працює коректно, або сама функція h не є достатньо стійкою. Факт що функція h не є стійкою поки не доведений, тож лишається лише не корекність алгоритму. При запуску брутфорс методу на найбільш вразливий вхід на функцію h, а саме $h(\alpha_1, \alpha_0, \alpha_0, \alpha_1)$, де α_0 нульове 32-бітне слово, а α_1 нульове 32-бітне слово тільки з 1 на найстаршому біті було виявлено 21 різних диференціалів, хоча метод видає лише 1 можливий. Метод брутфорс полягає в повному перебору всіх можливих 32-бітних слів як можливий γ, далі обчислюється його ймовірність, якщо вона не нульова, значить кандидат підходить. Звісно такий метод пошуку не годиться для використання до справжніх, а тільки в якості первірки некоректності методу. Навіть при тому, що метод повинен шукати тільки оптимальні диференціали цього не достатньо, оскільки часто таких диференціалів дуже мало і в подальшому такі диференціальні характеристики зникають.

Той факт, що підхід використовувати лише диференціальні характеритики з найвищою з ймовірностю повертає вибірку диференціальних характеритик розподіл ймовірності яких схожий до нормального розподілу не зовсім очевидний, проте підвердився на практичнці.

2.3 Аналіз алгоритму пошуку всіх дифренціальних характеристик

В розділі 1.2 в деталях описано алгоритм який повеннен повератати всі можливі оптимальні диференціальні характеристики. Нагадую, що це такі які мають найбільшу ймовірність. Проте в даному розділі наведено контрприклад роботи, а саме те що він не завжди видає всі необіхідні результати та навіть більше, що видає неможливі диференціли, тобто ті,

які мають нульову ймовірінсть.

Нехай $\alpha = \texttt{0b10000000}$, $\beta = \texttt{0b11000000}$, тоді $C(\alpha, \beta) = \texttt{0b00000000}$, необхдіно найти такі γ, що $xdp^+(\alpha, \beta \to \gamma) \neq =$. Застосуємо Алгоритм 1.3 покроково (перевірку на $p_i = 1$ та $i = n - 1$, пропусщено з очевидних причин):

1) $\gamma_0 = \alpha_0 \oplus \beta_0 = 1 \oplus 1 = 0$

2) γ_1: $\alpha_0 = \beta_0 \neq \gamma_0 \Rightarrow \alpha_1 \neq \beta_1 \Rightarrow \gamma \in \{0, 1\}$

3) γ_2: $\alpha_1 \neq \beta_1 = \gamma_1 \Rightarrow \alpha_2 = \beta_2 \Rightarrow \gamma_2 = \alpha_2 = 0$

4) γ_3: $\alpha_2 = \beta_2 = \gamma_2 \Rightarrow \gamma_3 = \alpha_3 \oplus \beta_3 \oplus \alpha_2 = 0 \oplus 0 \oplus 0 = 0$

5) γ_4: $\alpha_3 = \beta_3 = \gamma_3 \Rightarrow \gamma_4 = \alpha_4 \oplus \beta_4 \oplus \alpha_3 = 0 \oplus 0 \oplus 0 = 0$

6) для $\gamma_5 - \gamma_7$ аналогічно пункту 5)

В пункті 2) неважливо чи обирати 0 чи 1, оскільки результат в 3) залишиться незміним. Одже алгоритм в сумі згенерував 2 диференціали `0b00000000` та `0b01000000`. Брутфорс алгоритм також згенерував 2 диференціали `0b01000000` та `0b11000000`, проте як видно є одна розбіжність, тож при перевірці Алгоритмом 1.2 виявилось, що диференціали згенеровані бруфорс методом мають ймовірності по 0.5, тобто $xdp^+(\texttt{0b10000000}, \texttt{0b11000000} \to \texttt{0b01000000}) = 0.5$ та $xdp^+(\texttt{0b10000000}, \texttt{0b11000000} \to \texttt{0b11000000}) = 0.5$ в свою чергу $xdp^+(\texttt{0b10000000}, \texttt{0b11000000} \to \texttt{0b00000000}) = 0$, що фактично очевидно.

Із запропонованого контрприкладу можна чітко стверджувати, що Алгоритм 1.3 не є повністю коректиним, через те може видадавати не коректні вихідні диференціали, що відповідно вимагає додаткової валідації даних. Більше того, алгоритм може не згенерувати жодного ненульового диференціала як наприклад при вході $\alpha = \beta = \texttt{0xC0}$ чи $\alpha = \texttt{0xA0}, \beta = \texttt{0x60}$, проте інколи також може і згенерувати всі можливі диференціали.

На основі вищесказаного, Алгоритм 1.3 не рекомендується використовувати як основний метод пошуку диференціальних характеристик через його низьку ефективність та обмежену гнучкість, а

точніше її відсутність як такої.

На рисунках 2.3, 2.4 та 2.5 продемостровано прорівняння результатів роботи алгоритму брутфорсу та Алгоритму 1.3. Цей рисунок є прямим доказом, що алгоритм далеко не завжди працює коректно і часто може не видавати коректних результатів. Генерування відбувалось на всіх можливих 4-бітних словах, тобто $len(\alpha) = len(\beta) = 4$ і сумі виходить $2^4 \cdot 2^4 = 2^{4+4} = 256$ можливих комбінацій α та β. Лінія «Загальний BF» показує кількість всіх можливих диференціальних характеристик для відповідної пари (α β) і відовідно лінія «Фільтрований BF» показує кількість диференціальних характеристик для яких ймовірність $max(xdp^+(\alpha, \beta \rightarrow \gamma))$ максимальна. Лінія «Загальний A3» показує кількість виходів, які були згенеровні Алгоритмом 1.3 і відповідно лінія «Фільтрований A3» показується кількість ненульових вихідних дифференціалів, через специфіку алгоритму очевидно, що вони всі оптимальні.

Спочтаку необіхдно зрозуміти як часто алгоритм та видає неможливі диференціальні характеристики, що продемостровано на Рисунку 2.3. Очевидно, що існує 3 основних сценаріїв, коли кількість спіпадає, спвпадає частоково та зовсім відсутні можливі виходи з алгоритму. Що очевидно з графіка кільсть повних співпаднь практично відсутня, проте кількість нульових виходів з алгоритму значна. У решти варіантів різниця в кращому з випадків у два рази.

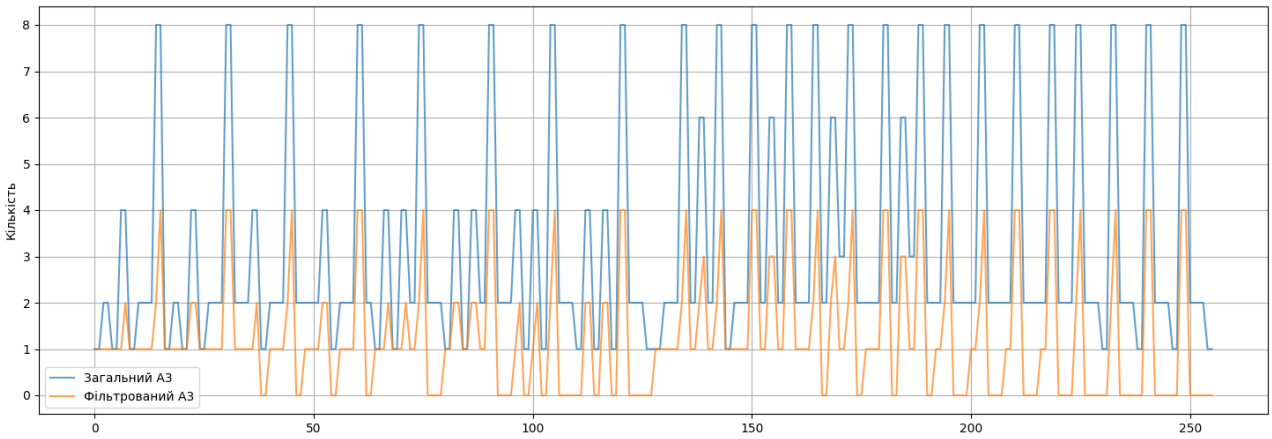


Рисунок 2.3 – Порівняння кількості всіх можливих виходів Алгоритму 1.3 для 4-бітних слів з відфільтованими ненульовими Алгоритму 1.3

Далі необхідно зрозуміти як багато алгоритм видає коректних виходів в порівняні зі всіма можливими. Отже на рисунку 2.4 це продемонстровано. Як видно практично не буває повних співадінь по розмірам, а в середньому різниця в два рази. Можна зробити висновок про ефектичність пошуку всіх оптимальних диференціальних характеристик алгоритму 1.3.

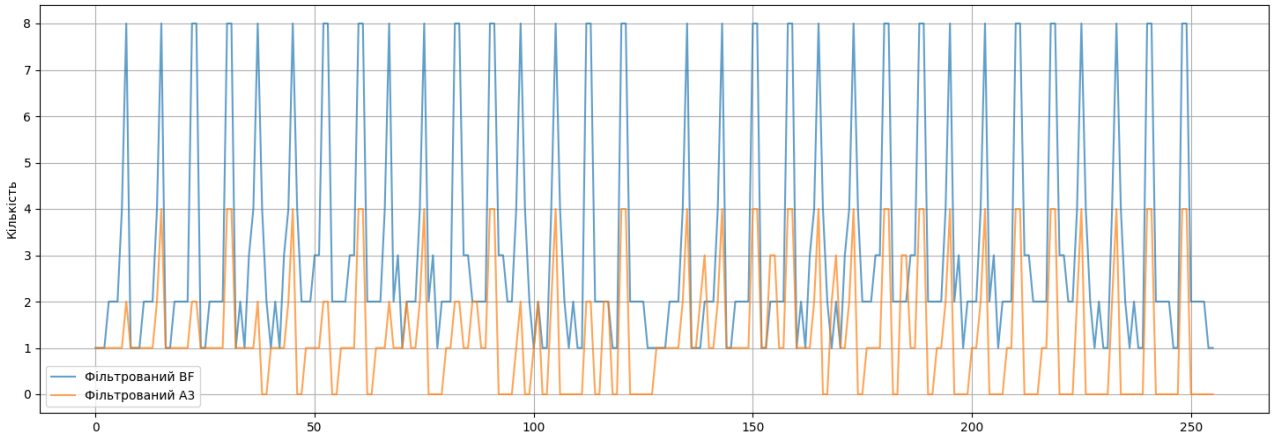


Рисунок 2.4 – Порівняння кількості всіх можливих оптимальних виходів з відфільтованими ненульовими Алгоритму 1.3

І в кінці важливо розуміти як багато втрачається всіх диференціалів, якщо використовувати 1.3. На рисунку 2.5 чітко

прослідковується закономірність того що в середньому кількість дифренціальних характеристик як мінімум в два рази менша ніж середня кількість всіх можливих.

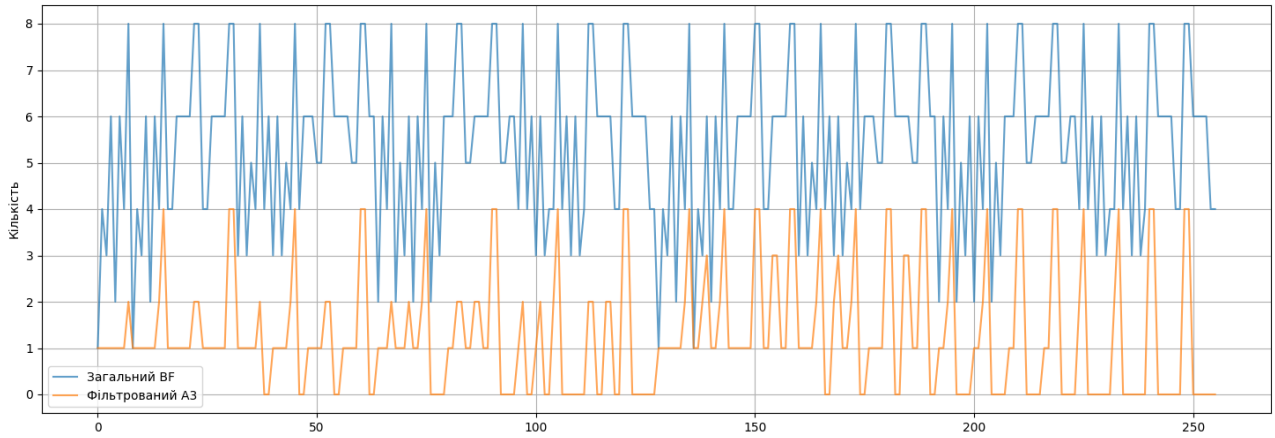


Рисунок 2.5 – Порівняння кількості всіх можливих виходів і відфільтрованого результату з Алгоритму 1.3 для 4-бітних слів

Висновок, Алгоритм 1.3 не рекомендується для використання через його непрактичність, нефективність (необхідність в додатковій перевірці) та відсутності хоча б мінімальної гнучкості.

2.4 Побудова фільтра для модифікованого «Кипарис-256»

В даному розділі описано застосування атаки Зустріч у Фільтрі на модифіковану версію блокового шифру «Кипарис-256». Як відомо з розділу 1.6 власне атаку можна розділіти на 3 етапи:

1) на вхід пара відкритих текстів (P_1, P_2) і різниця $\oplus\ \Delta_{IN} = P_1 \oplus P_2$
2) знайти r-раундовий дифренціал з ймовріністю p, $\Delta_{IN} \xrightarrow{p} \Delta_{OUT}$
3) знайти множину $\mathcal{S}$ 1.2
4) знайти множину $\mathcal{T}$ 1.3
5) найти співпадіння між множинами $\mathcal{S}$ та $\mathcal{T}$ 1.4

Після закінчення процедури такої фільтрації пари $(P_1,\ P_2)$ вона запам'ятовується хороша пара разом із всіма відповідними

диференціальними характеристиками і подається до процедури відновлення ключа.

В такій схемі необхідно мати розуміння декількох уточнючих питань, від цього залежать деталі реалізації та власне результати. А саме, які ймовірності краще обирати для $\mathcal{S}$ і $\mathcal{T}$, очевидно це залежить від самої множини, розповсюдження дифференціальних характеристик після кожного раунда та власне кількості раундів. Які конфігурації підійдуть найкраще для модифікованго «Кипарис-256», як їх обирати та які критерії можуть існувати. Скільки мінімально необхдіно раундів для майже гарантовного знаходження хорошої пари.

В даній роботі вибрана наступна конфігурація застосування Зустріч у Фільтрі. Для початку була обрана спрощена конфігурація, а саме нехай початкой r-раундовий диференціал відомий, далі застосовуємо комібнацію двух алгоритмів 1.3 1.4 (пошук всіх оптимальних та пошук одного оптимального), для підняття ефективності Алгоритму 1.3 (повинна бути як мінімум одна диференціальна характеристика, далі комбінований алгоритм). Таку комбінацію двух алгоритмів застосовуємо до функції і будуємо на цій основі раунд, тобто далі це пошук всіх оптимальних дифренціальних характеристик для одного раунда модифікованого «Кипарис-256». Після одного раунда на виході получаємо множину кандидатів, на цьому етапі можемо застосувати фільтрацію за ймовірністю p, тобто обрати мінімальний поріг чи обрати x кращих диференціальних характеристик з найбільшою йомвірністю. Для усіх виходів з процедури повторюємо її на на y раундів і формуємо фінальний результат, можливо варто пропустити проміжну фільтрацію на остаьому раунді. Таким чином будуємо множну $\mathcal{S}$ 1.2.

Тепер необхібно знайти множину $\mathcal{T}$ 1.3. Існує два очевидних способи або мати вже відому різницю Δ_C, після шифрування (P_1, P_2) або пройти ще додаткових y^* раундів такою ж процедурою і обрати в кінці випадковий дифереціал (можливо з найбільшою йморвіністю або декілька

найкращих). Таким чином відомо Δ_C, тепер враховуючи властивіть схеми Фейстеля про шифрування та розшифрування і те, що ці процеси ідентичні, але з єдиною відмінстю, вхідні параметри потрібно поміняти місцями, тобто $L_{t-1} = R_{t-1}, R_{t-1} = L_{t-1}$. Після того як входи на раунд помінялись місцями запускаємо аналогічний процес як для пошуку $\mathcal{S}$. І ось вже знайдено множину $\mathcal{T}$. Також варто пам'ятати про проміжну фільтрацію диференціальних характеристик, оскільки в даному випадку це більш, через неможливість передобчислення множини $\mathcal{S}$.

Фінальним етапом фільтрація Зустріч у Фільтрі являється пошуком співпадінь характеристик між множинами $\mathcal{S}$ та $\mathcal{T}$, за $\tau_u = (\tau_t || \tau_s) = \Delta C \xrightarrow{\text{u}} \Delta_{OUT} = \Big(\Delta C \xrightarrow{\text{t}} (\Delta Y = \Delta X) \xrightarrow{\text{s}} \Delta_{OUT}\Big)$.

Блоковий шифр «Кипарис» побудований на схемі Фейстеля, за структорою шифру, вхідне слово розбивається на дві частини, потім ліва частина $\oplus$ з раундовим ключем і знову розбиваєтья на 4 s-бітових слова. В розділі 2.2 найдені декільки вхідних диференціалів, які провокують меншу кількість диференціальних характеристик функції h, далі вхідні диференціали підібрані з цим врахуванням. Після кожного раунда (окрім останього) відбирається x диференцільних характеристик за найвищими ймовірностями.

З врахуванням результатів атаки на $CHAM$ [3] до «Кипарис-256», потрібно оцінити зростання кількості диференціальних характеристик після кожного раунда, щоб сформувати риблизне уявлення скільки мінімально раундів необхідно для знаходження хорошої пари для фільтра. В контексті фільтра Зустріч у Фільтрі це просто необхдіно для його мінімальної роботоздатності, тобто щоб гарантовано знаходилось як мінімум одне співпадіння між множинами. Особливо важливо це знати для розміння кількісті нижніх раундів, бо кількість верхніх можна передобчислити і відповідно взяти значну кількість. Далі попередньо оцюнюється чи комібніація двух алгоритмів буде повертати достатню кількість диференціальних характеристик для побудови атаки Зустріч у

Фільтрі.

Вхідне слово	**Значення**
1	$(\texttt{C0000000 00000000 00000000 C0000000})_{16}$
2	$(\texttt{00000000 00000000 C0000000 00000000})_{16}$
3	$(\texttt{C0000000 C0000000 00000000 00000000})_{16}$
4	$(\texttt{E0000000 00000000 00000000 E0000000})_{16}$
5	$(\texttt{00000000 00000000 E0000000 00000000})_{16}$
6	$(\texttt{E0000000 E0000000 00000000 00000000})_{16}$

Таблиця 2.3 – Таблиця вхідних слів для таблиці 2.4

В таблиці 2.4 продемостровано як швидко зростає кількість диференціаліва після кожного раунда з відподвіною комбінацією вхідних слів з таблиці 2.3.

	Вхід-1	Вхід-2	Вхід-3	Вхід-4	Вхід-5	Вхід-6
Раунд 1	2	3	3	5	10	14
Раунд 2	5	6	8	148	3100	32
Раунд 3	22	15	21	75937	x	7908
Раунд 4	194	1967	24	x	x	x
Раунд 5	21728	x	6148	x	x	x

Таблиця 2.4 – Кількість вихідних диференціалів на кожному раунді

Висновок з результатів в таблиці 2.4, можна впевнено сказати, що такої кількості диференціалів не достатня, оскільки для отримання достатньої кількості необхідно пройти більше раундів ніж попередньо очікувалось, тож вже очевидно, що даний підхід не працює і варто переглянути стратегію пошуку диференціальних характеристик на ефективнішу.

Можна ще стверджувати, що брати вибіркові вхідні диференціали не зовсім коректний підхід, проте це зручний ручний спосіб контролювати кількість диференціальних характеристик на першому раунді, але в наступних раундах це непередбачувано особливо для різних

нижніх раундів тому, що Δ_{OUT} обчислюється динамічно. Нижче це явно продемостровано, що навіть при випадкових входах (як вхідну різницю обиралось результат шифрування псевдовипакового слова), кількість диференціалів може бути нехтовно малою. Нехай на вході два слова (P_1, P_2) в яких різниця (1) (шістнадцяткове представлення) $\Delta_{IN} =$ (`5684C6C6 CAA0A5D3 41E8B18A 5684C6C6 A85D3875 C187AD2A 5684C6C6 563288FB`) для таблиці 2.5 і друга вхідна різниця для таблиці 2.6 $\Delta_{IN} =$ (`A4513875 C987AD2A 5684C6C6 563280FF 46465968 23EF0FB4 E04B74C6 AA05076B`). Зашифровуємо відповідно $P_1, P_2 \rightarrow C_1, C_2$, отримуємо $\Delta_{OUT} = C_1 \oplus C_2$. Очевидно, що Δ_{OUT} напряму залежить від кількості раундів.

Конфігурація фільтра (s,t)	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{T}\rvert$
(1 - 1)	$2^{6.01}$	$2^{6.01}$
(1 - 2)	$2^{6.01}$	$2^{8.01}$
(2 - 1)	$2^{14.07}$	2^{0}
(2 - 2)	$2^{14.07}$	$2^{9.03}$
(2 - 3)	$2^{14.07}$	$2^{20.06}$

Таблиця 2.5 – Приклад вхідної різниці 1

Конфігурація фільтра (s,t)	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{T}\rvert$
(1 - 1)	2^{0}	2^{0}
(1 - 2)	2^{0}	2^{0}
(2 - 1)	2^{0}	2^{0}
(2 - 2)	2^{0}	$2^{8.08}$
(2 - 3)	2^{0}	2^{0}

Таблиця 2.6 – Приклад вхідної різниці 2

В таблицях 2.6 2.5 продемонстровано, що навіть при псевдовипадкових входах результати фільтра не втішні та розповсюдженняя, або незначне або зовсім відсутнє. В усіх випадках із таблиць співдадінь не спостергігалось, також не найшлось жодної конфігурації та входів, де б зустрічалось хоча б одине співпадіння між множинами, не тільки в продемострованих випадках, а і в багатьох інших які тут явно не вказані. Це говорить про стійкість до фільтра з вище

описаної будовою.

Підбиваючи підсумки до цього розділу, використовавати алгоритм 1.3 в якості основного методу пошуку дифреенціальних характеристик виявилось недоцільно. Причинии тому дві, по-перше за своєю природою алгоритм повертає лише оптимальні диференціали, що сильно обмежує його ефективність, особливо при великих n, по причинам характеру розподілу $xdp^+(\alpha, \beta \rightarrow \gamma)$. Друга причина, що алгоритм повертає в середньому половину оптимальних диференціалів і навіть використання комбінованого алгоритма (тільки він і використовувався для фільтра) недостатньо. Через це фільтр в купі з різними конфігураціями повертають недостатньо диференціальів для множин $\mathcal{S}$, $\mathcal{T}$. Така будова фільтра не працює для модифікованого шифру «Кипарс-256» і відповідно шифр стійкий до такого підходу.

Висновки до розділу 2

У цьому розділі було імплементовано та досліджено алгоритм пошуку всіх оптимальних дифренціалів. Було виявлено та наведено контрприклад несправності цього алгоритму та його практичну незастосовність через необхідність додаткової перевірки вихідних диференціалів та значну втрату можливих вихідних диференціалів. Це фактично унеможливлює його самостійне використання та вимагає додаткових алгоритмів. Як приклад, було використано алгоритм пошуку одного оптимального диференціала. Таким чином, мінімально необхідно використовувати комбінований підхід, який поєднує обидва ці способи пошуку.

Розроблено загальний алгортим пошуку диференціальних характеристик функції h через пошуку диференціалів xdp^+. Застовано вище сказаний комібінований алгоритм пошуку оптимальних дифрернціальних характеристик до розробленого алгоритму до функції

h. Проаналізовано отримані результати та виявлено закономірнсті та потенційні слабкі місця у функції h.

Застосовано метод Зустріч у Фільтрі для побудови фільтра до модифікованого шифру «Кипарис-256» та проаналізовано стійкість шифру «Кипарис-256» до даного методу диференціального аналізу.

ВИСНОВКИ

У ході дослідженя було отрмано наступну низку практичних результатів:

1. Проведено огляд класичних робіт по диференціальному аналізу аналізу, в тому числі для ARX-криптосистем. Оглянуто такі блокові шифри як $SPECK$, $CHAM$ та «Кипарис» їхні конфігурації та особливості. Розглянуто як класичні так і сучасні алгоритми пошуку диференціальних характеристик за оперціює модульного додавання.

2. Проаналізовано метод Зустріч у Фільтрі та його застосування до наступних блокових шифрів $SPECK$ та $CHAM$ із акцентом на пошук диференціальних характеристик. Переглянуто спільні механіки та принципи побудови фільтрів, а також статичстичні особливості результатів та конфігурації метода Зустріч у Фільтрі.

3. Запропоновано алгоритм побудови диференціальних характеристик і фільтра для малоресурсного шифра «Кипарис». В ході побудови та дослідження алгоритму виявлено, що класичні алгоритми для пошуку оптимальних диференціалів, які були взяті за основу мають обмеження та не видають бажаного результату. В свою чергу це змусило використовувати комбінований підходу використання двух алгоритмів для їх ефективного використання, але виявлено, що такий підхід також мало ефективний.

4. Проаналізовано застосовність метода Зустріч у Фільтрі для модифікованого шифру «Кипарис». Виявлено, що метод не є застосовним до шифру «Кипарис», в контексті використання побудованого комбінованого алгоритму генерування диференціальних характеристик, оскільки кількість вихідних диференціалів за попередніми спробами та оцінками не є достатньою для ефективного використання фільтра.

Напрямки подальших досліджень:

1. Побудувати покращений алгоритм пошуку диференціальних характеристик для шифру «Кипарис»

2. Оцінити швидкість зростання дифренцільних характеристик та оцінити мінімально необхідну кількість раундів для успішного застосування фільтра та відповідно атаки Зустріч у Фільтрі.

3. Спробувати різні підходи до фільтрації дифренціалів, як на основі ймовірносетй так і на основі слабких місць гілок функції h.

ПЕРЕЛІК ПОСИЛАНЬ

[1] Ray Beaulieu та ін. *The SIMON and SPECK Families of Lightweight Block Ciphers*. Cryptology ePrint Archive, Paper 2013/404. https://eprint.iacr.org/2013/404. 2013. URL: https://eprint.iacr.org/2013/404.

[2] Eli Biham та Adi Shamir. «Differential Cryptanalysis of DES-like Cryptosystems». В: *J. Cryptology* 4 (1991), с. 3—72. DOI: 10.1007/BF00630563.

[3] Alex Biryukov, Je Sen Teh та Aleksei Udovenko. «Advancing the Meet-in-the-Filter Technique with Applications to CHAM and KATAN». В: *IACR Cryptology ePrint Archive* 2023.851 (2023). [Електроний ресурс]. URL: https://eprint.iacr.org/2023/851.pdf.

[4] Alex Biryukov та ін. «Meet-in-the-Filter and Dynamic Counting with Applications to Speck». В: *IACR Cryptology ePrint Archive* 2022.673 (2022). [Електроний ресурс]. URL: https://eprint.iacr.org/2022/673.pdf.

[5] Mingjiang Huang та Liming Wang. *Automatic Tool for Searching for Differential Characteristics in ARX Ciphers and Applications (Full Version)*. Cryptology ePrint Archive, Paper 2019/1318. https://eprint.iacr.org/2019/1318. 2019. URL: https://eprint.iacr.org/2019/1318.

[6] Helger Lipmaa та Shiho Moriai. «Efficient algorithms for computing differential properties of addition». В: *IACR Cryptology ePrint Archive* 2001.001 (2001). [Електроний ресурс]. URL: https://eprint.iacr.org/2001/001.pdf.

[7] Byoungjin Seok та Changhoon Lee. «Fast implementations of ARX-based lightweight block ciphers (SPARX, CHAM) on 32-bit processor». В: *International Journal of Distributed Sensor Networks* 15 (2019). URL: https://api.semanticscholar.org/CorpusID:203168876.

[8] Марія Юріївна Родінко. «Методи побудови та дослідження властивостей малоресурсних блокових шифрів та їх компонентів: дисертація... доктора філософії за спеціальністю 122 – комп'ютерні науки (12 – Інформаційні технології)». В: (2020), 201 с.