

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

«На правах рукопису»

УДК 519.719.2

«До захисту допущено»

Завідувач кафедри

_____ Сергій ЯКОВЛЄВ

«__» _____ 2026 р.

Дипломна робота
на здобуття ступеня бакалавра
за освітньо-професійною програмою
«Математичні методи криптографічного захисту інформації»
зі спеціальності: 113 Прикладна математика
на тему: «Методи криптоаналізу Grøstl-подібних
геш-функцій»

Виконав:

студент IV курсу, групи ФІ-23

Гутовський Матвій Андрійович

Керівник:

зав. кафедри ММЗІ, к.т.н.

Яковлев Сергій Володимирович

Рецензент:

доцент кафедри ІБ, к.т.н., доцент

Стьопочкіна Ірина Валеріївна

Засвідчую, що у цій дипломній
роботі немає запозичень з праць
інших авторів без відповідних
посилань.

Студент _____

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

«__» _____ 2026 р.

ЗАВДАННЯ
на дипломну роботу

Студент: Гутовський Матвій Андрійович

1. Тема роботи: «*Методи криптоаналізу Grøstl-подібних геш-функцій*», науковий керівник дисертації: зав. кафедри ММЗІ, к.т.н. Яковлєв Сергій Володимирович,

затверджені наказом по університету №__ від «__» _____ 2026 р.

2. Термін подання студентом роботи: «__» _____ 2026 р.

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

4. Предмет дослідження: Методи побудови триангуляційних атак на геш-функції

5. Перелік завдань:

– провести порівняльний аналіз архітектур фіналіста конкурсу SHA-3 Grøstl та національного стандарту ДСТУ 7564:2014 «Купина», зокрема специфікацій їхніх функцій стиснення;

– формалізувати модель застосування алгоритму триангуляції (Super-Inbound фази) для пошуку колізій в геш-функціях з AES-подібним перетворенням;

3

– дослідити процес поширення усічених диференціалів у нелінійній гілці T^+ алгоритму «Купина», дослідити процес та вплив операції модульного додавання $\boxplus_{64}$ на руйнування диференціальних шляхів;

– Оцінити складність та порівняльний аналіз просторової й обчислювальної складності атаки для редукованих раундів оригінальної та модифікованої (без переносів) архітектур стандарту.

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

7. Орієнтовний перелік публікацій: планується доповідь на всеукраїнській конференції

8. Дата видачі завдання: 10 вересня 2025 р.

Календарний план

№ з/п	Назва етапів виконання кваліфікаційної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2025 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2025 р.	Виконано
3	Порівняльний аналіз архітектур ген-функцій Grøstl та ДСТУ 7564:2014 «Купина»	Листопад-грудень 2025 р.	Виконано
4	Формалізація математичної моделі триангуляційної атаки та дослідження гілки T^+	Січень-лютий 2026 р.	Виконано
5	Розробка програми мовою Python та проведення емпіричних експериментів	Березень 2026 р.	Виконано
6	Розрахунок просторової й обчислювальної складності триангуляційної атаки на модифіковану «Купину»	Квітень 2026 р.	Виконано
7	Написання тексту пояснювальної записки та оформлення кваліфікаційної роботи	Квітень-травень 2026 р.	Виконано
8	Отримання відгуку керівника, рецензії та підготовка презентації до захисту	Травень-червень 2026 р.	Виконано

Студент _____ Матвій ГУТОВСЬКИЙ

Керівник _____ Сергій ЯКОВЛЄВ

РЕФЕРАТ

Кваліфікаційна робота містить: 55 стор., 12 рисунків, 9 таблиць, 17 джерел.

Метою кваліфікаційної роботи є дослідження криптографічної стійкості алгоритму «Купина» до методів алгебраїчного криптоаналізу, зокрема до триангуляційної атаки рикошетом. Об'єктом дослідження є криптографічна стійкість геш-функцій з AES-подібними перетвореннями до методів алгебраїчного криптоаналізу. Предметом дослідження є методи побудови триангуляційної атаки на функцію стиснення національного стандарту ДСТУ 7564:2014 «Купина» та оцінка впливу операції арифметичного додавання за модулем на обчислювальну складність.

У ході виконання роботи було проведено порівняльний аналіз архітектур геш-функцій Grøstl та «Купина». Формалізовано математичну модель фази атаки Super-Inbound та досліджено процес поширення усічених диференціалів у нелінійній гілці алгоритму. Для емпіричної верифікації ймовірнісних характеристик руйнування диференціальних шляхів було розроблено програмну реалізацію мовою Python.

Виконано розрахунок просторової та обчислювальної складності атаки. Показано, що для базової AES-подібної архітектури (без переносів) триангуляційна атака дозволяє побудувати диференціальний шлях на 7-й та 11-й раунди зі складністю 2^{72} та 2^{168} відповідно. Водночас математично обґрунтовано, що наявність операції модульного додавання в оригінальній архітектурі «Купина» генерує каскад нелінійних переносів, які унеможливають аналітичне розв'язання системи.

ГЕШ-ФУНКЦІЯ КУПИНА, GRØSTL, AES-ПОДІБНІ ПЕРЕТВОРЕННЯ, ТРИАНГУЛЯЦІЙНА REBOUND-АТАКА, REBOUND-АТАКА, ПОШУК КОЛІЗІЙ, УСІЧЕНІ ДИФЕРЕНЦІАЛИ, СТУПЕНІ СВОБОДИ, МОДУЛЬНЕ ДОДАВАННЯ

ABSTRACT

The qualification paper contains: 55 pages, 12 figures, 9 tables, 17 sources.

This thesis aims to investigate the cryptographic resilience of the «Kupyna» algorithm to algebraic cryptanalysis, specifically the ricochet triangulation attack. The subject of the study is the cryptographic resilience of hash functions with AES-like transformations against methods of algebraic cryptanalysis. The subject of the study is the methods for constructing a triangulation attack on the compression function of the national standard DSTU 7564:2014 «Kupyna» and the assessment of the impact of modular arithmetic addition on computational complexity.

During the course of the work, a comparative analysis of the architectures of the Grøstl and Kupyna hash functions was conducted. A mathematical model of the Super-Inbound phase of the attack was formalized, and the process of propagating truncated differentials in the nonlinear branch of the algorithm was investigated. A software simulator in Python was developed for empirical verification of the probabilistic characteristics of differential path destruction.

The spatial and computational complexity of the attack was calculated. It was shown that for a basic AES-like architecture (without carry operations), the triangulation attack allows constructing a differential path for 7 and 11 rounds with a complexity of 2^{72} and 2^{168} , respectively. At the same time, it has been mathematically proven that the presence of modular addition in the original «Kupyna» architecture generates a cascade of nonlinear carry-overs, which make an analytical solution of the system impossible.

KUPYNA HASH FUNCTION, GRØSTL, AES-LIKE TRANSFORMATIONS, TRIANGULATING REBOUND-ATTACK, REBOUND ATTACK, COLLISION SEARCH, TRUNCATED DIFFERENTIALS, DEGREES OF FREEDOM, MODULAR ADDITION

ЗМІСТ

Перелік умовних позначень, скорочень і термінів	8
Вступ.....	10
1 Огляд атак на геш-функції, що використовують AES-подібні перетворення	11
1.1 Grøstl-подібні геш-функції	11
1.2 Геш-функція Grøstl	12
1.3 Геш-функція «Купина»	16
1.4 Порівняльний аналіз Grøstl та «Купини»	18
1.5 Теоретичний базис: Методи Rebound Attack.....	20
1.6 Хронологія та класифікація атак на Grøstl та «Купини»	24
2 Побудова триангуляційної атаки на геш-функцію «Купина»	30
2.1 Аналіз застосовності триангуляційних атак до алгоритму «Купина»	31
2.2 Формальна модель перенесення триангуляційної атаки на модифіковану Купину	39
2.3 Аналіз лінійного перетворення та властивостей MDS-матриць....	45
Висновки	47
Перелік посилань	49
Додаток А Великі рисунки та таблиці	51

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

$0x$	Префікс чисел, записаних у шістнадцятковій системі числення;
$GF(2^8)$	Скінченне поле (поле Галуа) з незвідним поліномом $f(x) = x^8 + x^4 + x^3 + x^2 + 1$;
$\oplus$	Логічна операція «виключне АБО» (XOR) для двійкових векторів (побітове додавання за модулем 2);
$\otimes$	Векторне множення над скінченним полем;
$\ll$	Логічний зсув вліво послідовності фіксованої довжини (у бік старших розрядів); молодші розряди заповнюються нулями; кількість позицій зсуву визначається другим аргументом;
$\gg$	Циклічний зсув (обертання) вправо послідовності фіксованої довжини (молодші розряди переміщуються на позиції старших);
$\kappa_\nu^{(l)}$	Функція додавання ітераційної константи до внутрішнього стану за модулем 2 (побітове додавання, XOR);
$\eta_\nu^{(l)}$	Функція додавання ітераційної константи до внутрішнього стану за модулем 2^{64} (арифметичне додавання елементів стовпчиків);

π'	Шар нелінійного бієктивного відображення (шар S-блоків), що реалізує підстановку байтів;
π_l	Перестановка елементів $g_{i,j} \in GF(2^8)$ внутрішнього стану (циклічний зсув рядків вправо);
ψ	Лінійне перетворення (множення вектора на матрицю над скінченним полем);
H	Геш-функція, визначена в цьому описі;
$H(IV, M)$	Геш-код повідомлення M ;
IV	Вектор ініціалізації (Initialization Vector);
n	Позначатиметься довжина геш-коду в бітах ($n = 8 \cdot s$, де $s \in \{1, 2, \dots, 64\}$).
<i>Купина</i>	геш-функція Купина

ВСТУП

Надійні геш-функції є фундаментальними компонентами сучасної криптографічної інфраструктури, забезпечуючи цілісність даних у цифрових підписах, кодах автентифікації повідомлень. Після виявлення теоретичних та практичних вразливостей в алгоритмах сімейств MD5 та SHA-1 криптографічна спільнота зосередилася на розробці та аналізі стандартів нового покоління. Особливу увагу привертають геш-функції з AES-перетвореннями, що успадкували перевірені часом структурні принципи від стандарту шифрування AES. До цього класу належать такі відомі алгоритми, як фіналіст конкурсу SHA-3 **Grøstl** та чинний стандарт України **Куруна**. Аналіз стійкості цих алгоритмів до новітніх методів криптоаналізу є критичним для оцінки безпеки криптографічної інфраструктури, на якій вони покладаються.

Grøstl є одним із 14 кандидатів другого раунду конкурсу NIST SHA-3 [5]. Той факт, що Grøstl успішно пройшов відбір до другого раунду конкурсу NIST, свідчить про його здатність протистояти широкому спектру атак. Це підтверджує значний запас міцності алгоритму, що відповідає сучасним вимогам безпеки.

Геш-функція Куруна була обрана новим українським стандартом ДСТУ 7564:2014 у 2015 році [13]. Вона призначена для заміни старого стандарту ГОСТ 34.311-95. Основними вимогами до неї були як високий рівень безпеки, так і хороша продуктивність програмної реалізації на 64-бітних процесорах загального призначення.

Для глибокого розуміння стану безпеки цих важливих криптографічних стандартів необхідно детально розглянути вже відомі дослідження криптоаналітичних методів, які будуть обговорюватися далі.

1 ОГЛЯД АТАК НА ГЕШ-ФУНКЦІЇ, ЩО ВИКОРИСТОВУЮТЬ AES-ПОДІБНІ ПЕРЕТВОРЕННЯ

Цей розділ присвячений огляду попередніх результатів за тематикою дослідження. Буде розглядатися багато різних атак на представників сімейства геш-функцій, які використовують AES-подібні перетворення, і детальніше — теорія rebound-атак. Представниками цього класу є Grøstl та український стандарт гешування Куруна, побудований на основі перетворень блокового шифру Kalyna, що також має AES-подібну архітектуру. Така конструктивна схожість робить їх потенційно вразливими до спільних класів атак.

1.1 Grøstl-подібні геш-функції

Геш-функцію з AES-перетворенням називають криптографічною геш-функцією, функцією стискання, яка використовує як складові елементи раундів шифрування AES (Advanced Encryption Standard), такі як заміна байтів за таблицею (S-Box), зсув рядків та перемішування стовпців (MDS-матриця).

- **SubBytes (SB)**: нелінійний стан заміни байтів за допомогою S-box;
- **ShiftRows (SR)**: перший етап нелінійного перетворення що циклічно зсуває рядки внутрішнього стану;
- **MixColumns (MC)**: другий етап нелінійного перетворення, що змішує байти в межах кожної колонки, зазвичай шляхом множення на MDS-матрицю;
- **AddRoundKey (AK)**: додавання раундового ключа до внутрішнього стану.

У контексті геш-функцій, заснованих на таких перестановках (англ. fixed permutations), раундовий ключ може бути відсутнім або заміненим на

додавання констант. Використання широких перестановок (тобто перестановок, що оперують великими блоками даних, наприклад, 512 або 1024 бітів) замість класичних блокових шифрів є характерною рисою таких геш-функцій, як Grøstl.

Геш-функції, що належать до класу Grøstl-подібних, є представниками ширшої родини геш-функцій з AES-перетвореннями. Їхня архітектура натхненна стандартом шифрування AES, що забезпечує високу продуктивність та глибокий рівень аналізу безпеки. Сама структура раундової функції зображена нижче:

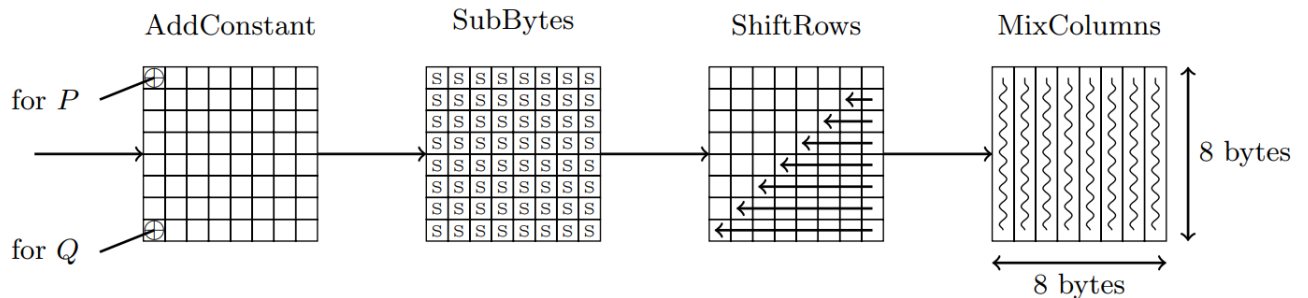


Рисунок 1.1 – AES-подібне перетворення [14]

1.2 Геш-функція Grøstl

У цьому розділі наводиметься опис **Grøstl**. **Grøstl** — це конструкція з широким каналом, яка використовує режим ланцюгового з'єднання, подібний до ітеративної схеми Меркла-Дамгора [2, 12, 10], де розмір внутрішнього стану значно більший за розмір виходу. **Grøstl** має хорошу продуктивність на широкому діапазоні різних платформ, а протидія атакам через бічні канали добре зрозуміла з подібних робіт над AES. Це все призводить до того, що всі відомі загальні атаки на геш-функцію стають набагато складнішими. Більш детальний опис самої Grøstl наведено у джерелах [11], [10].

Доповнення повідомлення

Як зазначають розробники **Grøstl** [5], довжина кожного блоку повідомлення дорівнює ℓ . Щоб мати можливість оперувати вхідними даними змінної довжини, визначено функцію доповнення **pad**. Ця функція доповнення приймає рядок x довжиною N бітів і повертає доповнений рядок $x^* || \text{pad}(x)$, довжина якого є кратною ℓ .

Функція доповнення виконує наступні дії. Спочатку до x додається біт 1. Потім додається $w = -N - 65 \bmod \ell$ бітів 0 і додається 64-бітне представлення числа $(N + w + 65)/\ell$. Це число є цілим завдяки вибору w , і воно представляє кількість блоків повідомлення у фінальному доповненому повідомленні.

Оскільки має бути можливим закодувати кількість блоків повідомлення у доповненому повідомленні в межах 64 бітів, максимальна довжина повідомлення на 65 бітів менша за довжину $2^{64} - 1$ блоків повідомлення.

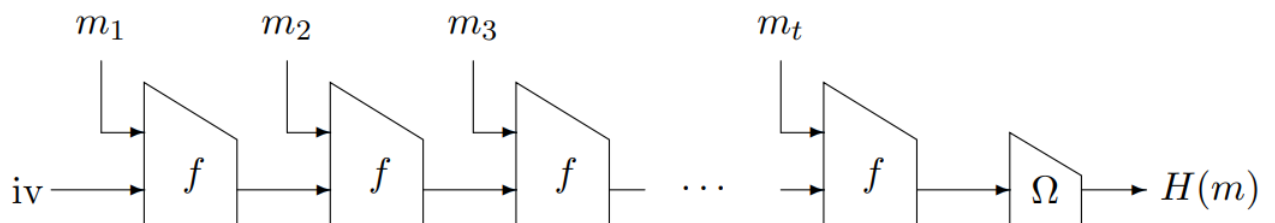


Рисунок 1.2 – Структура Grøstl [1]

Загальні параметри

Як показано на загальній схемі побудови Grøstl, повідомлення обробляється ітеративно. Розмір блоку повідомлення ℓ залежить від обраного варіанту функції:

- Для варіантів, що повертають до 256 біт, $\ell = 512$;
- Для більших варіантів $\ell = 1024$.

Процес перетворення стану

Після ініціалізації внутрішнього стану H_0 (англ. initial value, IV) та доповнення повідомлення, процес гешування відбувається ітеративно. Ітерація i оновлює ℓ -бітову змінну ланцюгового з'єднання H_i за допомогою ℓ -бітового блоку повідомлення M_i .

Цей процес описується застосуванням функції стиснення f :

$$H_i \leftarrow f(H_{i-1}, M_i) \quad \text{для } i = 1, \dots, t. \quad (1.1)$$

Функція f відображає два входи по ℓ бітів кожен (попередній стан H_{i-1} та блок повідомлення M_i) в один вихід довжиною ℓ бітів. Перший вхід називається *ланцюговим входом* (chaining input), а другий — *блоком повідомлення*.

Функція стиснення

Функція стиснення f побудована на основі двох подібних паралельних перестановок P і Q . Ці перестановки розроблені з використанням стратегії широкого каналу (англ. wide trail design strategy) і запозичують компоненти з AES, відрізняючись лише константами.

Оновлення ланцюгової змінної H блоком повідомлення M відбувається за формулою:

$$f(H, M) = P(H \oplus M) \oplus Q(M) \oplus H. \quad (1.2)$$

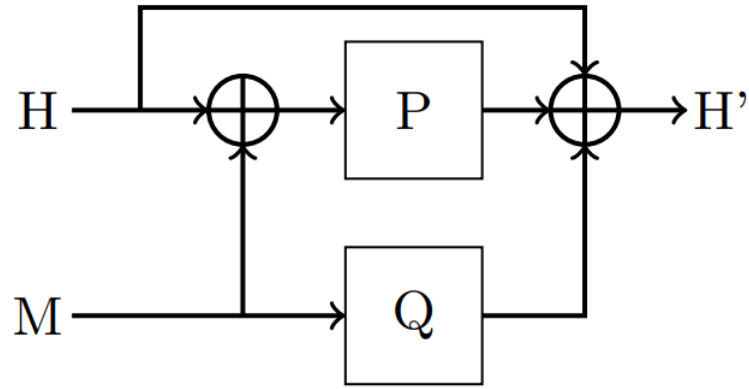
Знизу зображено загальний вигляд функції стиснення [14]:

Вихідне перетворення

Після обробки всіх t блоків повідомлення до останньої ланцюгової змінної H_t застосовується *вихідне перетворення* Ω , щоб отримати фінальний геш-код $H(M)$:

$$H(M) = \Omega(H_t). \quad (1.3)$$

Функція Ω визначена як відсікання (truncation) результату операції



Grøstl

Рисунок 1.3 – Функція стиснення Grøstl [14]

XOR між станом H_t та його перестановкою $P(H_t)$:

$$\Omega(H_t) = \text{trunc}_n(P(H_t) \oplus H_t), \quad (1.4)$$

де $\text{trunc}_n(x)$ — функція, що повертає останні n бітів рядка x . Це дозволяє отримати геш-код потрібної довжини n (наприклад, 256 або 512 бітів).

Дві перестановки P та Q побудовані з використанням стратегії широкого сліду і запозичують компоненти з AES. Дизайн цих двох перестановок дуже схожий на блоковий шифр W [11], що використовується у Whirlpool, реалізований з фіксованим вхідним ключем. Обидві перестановки оновлюють стан розміром 8×8 (64 байти) за 10 раундів кожна (16×8 за 14 раундів). Раундові перетворення дуже схожі на раундові перетворення AES і коротко описані в 1.1.

У кожному раунді стан оновлюється раундовим перетворенням r_i наступним чином:

$$r_i \equiv MB \circ ShB \circ SB \circ AC \quad (1.5)$$

1.3 Геш-функція «Купина»

«Купина» (або Куруна) [13] — це сімейство ітераційних геш-функцій, визначених в українському стандарті ДСТУ 7564:2014.

Принципи проектування Куруна дуже схожі на фіналіста SHA-3 Grøstl [5]. Як і в Grøstl, функція стиснення Куруна побудована на двох різних перестановках, які (до певної міри) базуються на принципах проектування AES. Далі ми опишемо компоненти геш-функції більш детально.

Доповнення повідомлення Геш-функція приймає на вхід повідомлення (бітовий рядок) довжиною N бітів. Кожне повідомлення доповнюється незалежно від його довжини. Процедура доповнення полягає у додаванні одного біта '1', потім d нульових бітів, де $d = (-N - 97) \bmod l$, та 96 бітів, що містять довжину повідомлення N (молодші біти у представленні довжини повідомлення мають менші індекси, тобто використовується порядок little-endian). У результаті довжина доповненої послідовності стає кратною довжині внутрішнього стану l , де $l \in \{512, 1024\}$. Максимальна довжина повідомлення обмежена $2^{96} - 1$ бітом.

Загальні параметри

Геш-функція H є залежним від IV відображенням повідомлення M у геш-код $H(IV, M)$, таким що:

$$\begin{aligned} H^{(IV)} : V_N &\rightarrow V_n, \quad n \in \{8 \cdot s \mid s = 1, 2, \dots, 64\}, \\ M &\in V_N, \quad N \in \{0, 1, \dots, 2^{96} - 1\}, \\ IV &\in V_l, \quad l \in \{512, 1024\}. \end{aligned}$$

Режим роботи геш-функції для $n \in \{8 \cdot s \mid s = 1, 2, \dots, 64\}$ позначається як **Куруна- n** . Основними рекомендованими режимами є **Куруна-256**, **Куруна-384** та **Куруна-512**.

Структура перетворення Н

Вхідне повідомлення для гешування завжди доповнюється (див. 1.3), щоб отримати довжину, кратну розміру блоку. Після доповнення воно розбивається на блоки $m_1, m_2, \dots, m_k$ довжиною l біт, де l визначається як:

$$l = \begin{cases} 512, & \text{якщо } 8 \leq n \leq 256, \\ 1024, & \text{якщо } 256 < n \leq 512. \end{cases} \quad (1.6)$$

Геш-код отримується відповідно до наступної ітераційної процедури:

$$\begin{aligned} h_0 &= IV, \\ h_\nu &= T_l^\oplus(h_{\nu-1} \oplus m_\nu) \oplus T_l^+(m_\nu) \oplus h_{\nu-1}, \quad \nu = 1, 2, \dots, k, \\ H(IV, M) &= R_{l,n}(T_l^\oplus(h_k) \oplus h_k), \end{aligned} \quad (1.7)$$

де:

$$IV = \begin{cases} 1 \ll 510, & \text{якщо } l = 512, \\ 1 \ll 1023, & \text{якщо } l = 1024, \end{cases} \quad IV \in V_l, \quad (1.8)$$

$T_l^\oplus, T_l^+$ — бієктивні відображення l -бітних блоків, а $R_{l,n}(X)$ — функція, що повертає n найбільш значущих бітів (MSB) з вхідного блоку X довжиною l (при $n < l$).

Перетворення $T_l^\oplus$ та T_l^+

Перетворення $T_l^\oplus$ та T_l^+ є бієктивними відображеннями $T_l^\oplus, T_l^+ : V_l \rightarrow V_l$, де $l \in \{512, 1024\}$. Кожне відображення реалізоване як ітеративне застосування кількох функцій, що приймають вхідний аргумент $x \in V_l$ у вигляді матриці байтів розміром $8 \times c$, представленої як елемент скінченного поля $GF(2^8)$.

Зв'язок між розміром внутрішнього стану (l), кількістю ітерацій (t), кількістю стовпців матриці стану (c) та довжиною геш-коду (n) наведено в таблиці 1.1.

Матриця внутрішнього стану позначається як $G = (g_{i,j})$, де $i = 0, 1, \dots, 7$, $j = 0, 1, \dots, c - 1$. Вона заповнюється вхідними байтами

$B_1, B_2, \dots, B_{l/8}$ у порядку стовпчик за стовпчиком (column-by-column).

Таблиця 1.1 – Загальні параметри для «Купини»

Довжина геш-коду (n)	Розмір внутрішнього стану (l)	Кількість раундів (t)	Стовпці матриці стану (c)
$8 \leq n \leq 256$	512	10	8
$256 < n \leq 512$	1024	14	16

Схематичне заповнення матриці стану виглядає наступним чином: вхідна послідовність байтів розподіляється по стовпцях, починаючи з першого ($g_{0,0} \dots g_{7,0}$), потім другий ($g_{0,1} \dots g_{7,1}$) і так далі до останнього байта стану.

Перетворення $T_l^\oplus$ та T_l^+ визначаються як добуток раундових функцій:

$$T_l^\oplus = \prod_{\nu=0}^{t-1} (\psi \circ \tau^{(l)} \circ \pi' \circ \kappa_\nu^{(l)}), \quad (1.9)$$

Аналогічно визначається перетворення T_l^+ :

$$T_l^+ = \prod_{\nu=0}^{t-1} (\psi \circ \tau^{(l)} \circ \pi' \circ \eta_\nu^{(l)}), \quad (1.10)$$

де кожна функція відповідає певному шару перетворень (нелінійному, перемішуванню тощо) у раунді ν .

1.4 Порівняльний аналіз Grøstl та «Купини»

Як зазначалося в розділі 1.3, український стандарт Кіруна базується на принципах, закладених у фіналісті конкурсу SHA-3 — функції Grøstl. Обидві функції належать до класу конструкцій з широким каналом (wide-pipe) та використовуюють перетворення на основі AES.

Незважаючи на архітектурну схожість, між ними існують важливі відмінності в деталях реалізації, спрямовані на підвищення

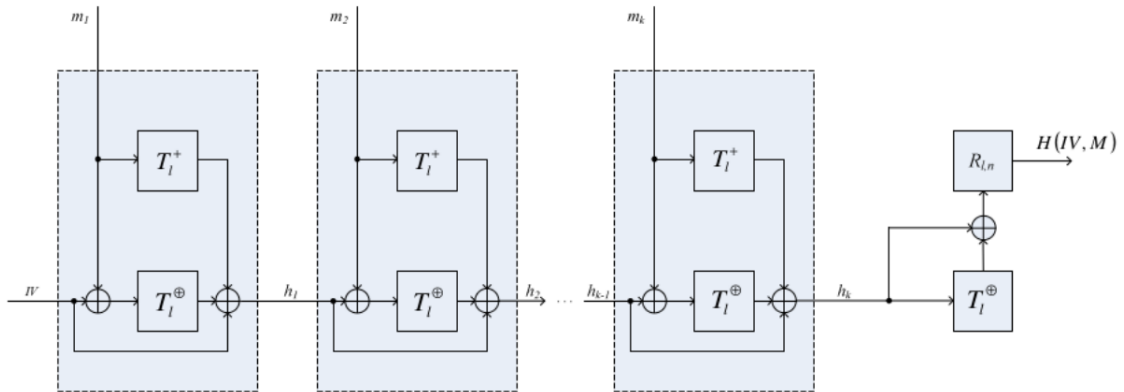


Рисунок 1.4 – Структура «Купини» [13]

криптографічної стійкості «Купини». Основні відмінності та спільні риси наведено в таблиці 1.2.

Таблиця 1.2 – Порівняльна характеристика Grøstl та «Купини»

Характеристика	Grøstl	«Купина»
Тип конструкції	Wide-pipe Merkle-Damgård	Wide-pipe Merkle-Damgård
Розмір блоку (l)	512 або 1024 біт	512 або 1024 біт
Кількість раундів	10 ($l = 512$) 14 ($l = 1024$)	10 ($l = 512$) 14 ($l = 1024$)
Padding	Додавання довжини у блоках (64 біти)	Додавання довжини у бітах (96 бітів)
Стиснення	$P(H \oplus M) \oplus Q(M) \oplus H$	$T^{\oplus}(H \oplus M) \oplus T^+(M) \oplus H$
Вихід	$\text{trunc}(P(H) \oplus H)$	$\text{trunc}(T^{\oplus}(H) \oplus H)$
Перестановки	P, Q (як у AES але різні)	$T^{\oplus}, T^+$ (як у AES але однакові)
S-Box	S-Box AES	Власні S-Box (4 шт.)
Поле Галуа $GF(2^8)$	$x^8 + x^4 + x^3 + x + 1$ (як у AES)	$x^8 + x^4 + x^3 + x^2 + 1$ (як у Калині)
Арифметичні операції	Лише побітове додавання $\oplus$ (XOR)	$\oplus$ та додавання за модулем 2^{64} (у перестановці T^+)
Формування IV	Нульовий блок, останні 16 біт кодують розмір вихідного гешу	Нульовий блок, нульовий байт якого кодує розмір вихідного гешу

Можна простежити пряму відповідність між компонентами функцій:

– Перестановка P у Grøstl функціонально відповідає перестановці $T^{\oplus}$ у Купині.

– Перестановка Q у Grøstl відповідає перестановці T^+ у Купині.

Рівняння функцій стиснення є алгебраїчно ідентичними. Якщо

позначити попередній стан як H_{prev} , а блок повідомлення як M , то для обох функцій оновлення стану виглядає як:

$$H_{new} = \mathbf{Perm}_1(H_{prev} \oplus M) \oplus \mathbf{Perm}_2(M) \oplus H_{prev}, \quad (1.11)$$

де $\mathbf{Perm}_1, \mathbf{Perm}_2$ — відповідні пари перестановок (P/Q або $T^\oplus/T^+$).

1.5 Теоретичний базис: Методи Rebound Attack

Rebound-атака [11] — це техніка ефективного використання доступних ступенів свободи.

Автори використовують усічені диференціальні шляхи, в яких більша частина витрат припадає на середні раунди. Потім, використовуючи локальну техніку типу «зустріч посередині» (meet-in-the-middle), ступені свободи витрачаються в середній частині диференціального шляху, саме там, де вони можуть найкраще поліпшити загальну складність. Точніше кажучи, деякі раунди всередині (так, скажімо, контрольовані раунди) будуть перевірятися в середньому лише декількома операціями, тоді як решта шляху як у прямому, так і в зворотному напрямку (неконтрольовані раунди) виконується, ймовірно. Цей криптоаналіз дає хороші результати [8] і може працювати без будь-яких спеціальних обмежень на диференціальному шляху. Однак контрольована частина обмежена двома раундами.

Основним результатом є впровадження методу криптоаналізу геш-функцій, який ми називаємо *rebound-атакою* (атакою рикошету). Ми застосовуємо його як до конструкцій на основі блокових шифрів, так і до конструкцій на основі перестановок. У rebound-атаці ми розглядаємо внутрішній шифр геш-функції або функції стиснення як три підшифри. Нехай E — блоковий шифр, тоді $E = E_{fw} \circ E_{in} \circ E_{bw}$. Альтернативно, для конструкції на основі перестановки ми розкладаємо перестановку P на три підперестановки $P = P_{fw} \circ P_{in} \circ P_{bw}$ [11].

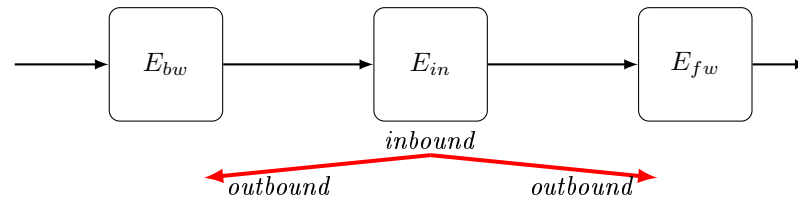


Рисунок 1.5 – Схематичне зображення rebound-атаки. Атака складається з однієї вхідної (inbound) та двох вихідних (outbound) фаз.

[11]

Rebound-атаку можна описати двома фазами (див. Рис. 1.5):

– **Вхідна фаза (Inbound phase):** Це фаза типу «зустріч посередині» (meet-in-the-middle) в E_{in} (або P_{in}), якій сприяють ступені свободи, доступні криптоаналітиці геш-функції. Поєднання техніки називається «зустріч посередині» та використання ступенів свободи, що призводить до дуже ефективних збігів, підходом **match-in-the-middle** (співпадиння посередині).

– **Вихідна фаза (Outbound phase):** На цій другій фазі використовується усічені диференціали як у прямому, так і у зворотному напрямках через E_{fw} та E_{bw} (або P_{fw} та P_{bw}), щоб отримати бажані колізії або псевдоколізії. Якщо усічені диференціали мають низьку ймовірність в E_{fw} та E_{bw} , можна повторити вхідну фазу, щоб отримати більше початкових точок для вихідної фази.

Розширення методу: Super-Inbound

На основі конференції CT-RSA 2009, де Ховратович, Бірюков і Ніколіч [7] представили інструмент «алгоритм триангуляції» — ефективний алгоритм на основі методу Гаусса для розв’язання системи біективних рівнянь, що дозволяє автоматично знаходити шлях до розв’язання нелінійної системи, у 2022 році [4] група дослідників (Сяо Дон, Кай Гао, Кетін Цзя і Сяюнь Ван) запропонувала суттєве вдосконалення класичного методу rebound-атак, яке отримало назву

Triangulating Rebound Attack (Триангуляційна атака рикошету). Сам алгоритм триангуляції моделює блоковий шифр типу AES як систему рівнянь розкладу ключів та функцій раундів, а байти стану та байти ключа — як змінні. Спочатку система визначається n початковими байтами стану та k початковими байтами ключа. Отже, коли m байтів є фіксованими, алгоритм повертає $n + k - m$ «вільних змінних» для формування базису системи. Алгоритм здатен точно вивести «вільні змінні», коли система є детерміновано розв'язною, та оптимізувати процес вгадування і перевірки цих змінних, коли система є ймовірно розв'язною.

Ідея триангуляційного алгоритму складається з наступних кроків:

- 1) Побудувати систему рівнянь, де змінними виступають байти стану. Заздалегідь відомі значення фіксуються як константи.
- 2) Усі змінні та рівняння позначаються як «необроблені».
- 3) Знайти змінну, яка входить лише в одне необроблене рівняння, та позначити її як «оброблену». Відповідне рівняння також позначається як «оброблене». Якщо такої змінної не існує, алгоритм перериває виконання цього циклу.
- 4) Якщо в системі залишаються необроблені рівняння, повернутися до кроку 3.
- 5) Усі змінні, що залишилися необробленими, визначаються як «вільні змінні».

Після ідентифікації масиву «вільних змінних» їм випадковим чином присвоюються значення, що дозволяє обчислити рішення для всієї нелінійної системи рівнянь. Класична rebound-атака, запропонована Менделем, зазвичай обмежується однією вхідною фазою (inbound phase), що покриває 1–2 раунди, де використовується техніка «зустріч посередині». Основною проблемою при розширенні атаки на більшу кількість раундів була складність узгодження (bridging) кількох незалежних вхідних фаз.

Основні ідеї модифікації [4]:

- **Алгебраїчний підхід:** Автори розглядають проблему з'єднання

кількох вхідних фаз як задачу розв'язання системи нелінійних рівнянь.

– **Використання ключів/констант:** Атака активно використовує ступені свободи (DoF), що надаються розкладом ключів (key schedule) або повідомленнями, які в попередніх атаках часто фіксувалися заздалегідь і не використовувалися ефективно.

– **Алгоритм триангуляції:** Для розв'язання отриманої системи рівнянь адаптовано алгоритм триангуляції (відомий з алгебраїчних атак), що дозволяє ефективно знаходити змінні та з'єднувати (bridge) кілька локальних частин атаки в одну глобальну структуру.

Це дозволило авторам покращити атаки на всі алгоритми, що мають AES або AES-подібну структуру, збільшивши кількість атакованих раундів або зменшивши складність існуючих атак.

Нижче наведено опис структури атаки, що базується на об'єднанні кількох фаз.

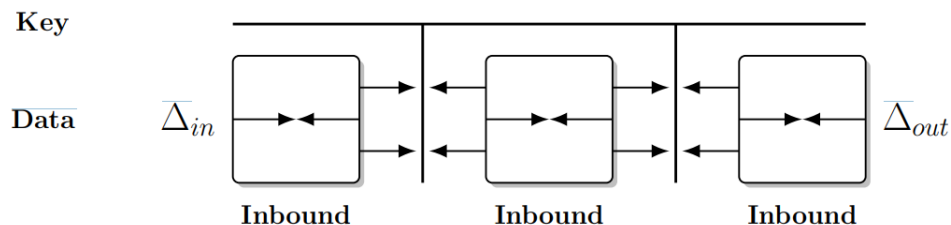


Рисунок 1.6 – Структура Superinbound [4]

Загалом, розвиток rebound-атаки можна розглядати як безперервний процес пошуку якомога більшої кількості ступенів свободи (DoF — Degrees of Freedom), щоб або зменшити загальну складність розв'язання вхідної фази, або розширити атаку на більше раундів. Як показано на рис. 1.6, кілька локальних вхідних частин утворюють *Super-Inbound* (супер-вхідну фазу).

Маючи вхідні/вихідні різниці $(\Delta_{in}, \Delta_{out})$ супервхідної фази, об'єднання кількох локальних вхідних частин полягає у пошуку узгодженої пари для $(\Delta_{in}, \Delta_{out})$, тоді як точні диференціали та значення

внутрішніх раундів знаходяться під контролем зломисника. Зазвичай $(\Delta_{in}, \Delta_{out})$ вибираються з усіченого диференціала rebound-атаки.

Підсумовуючи, джерелом DoF для всієї rebound-атаки може бути вибір:

- усіченого диференціала внутрішніх раундів Super-Inbound;
- вхідних/вихідних різниць та значень активних S-блоків стану;
- значень неактивних S-блоків стану;
- різниць байтів ключа;
- значень байтів ключа.

У цьому списку різниці та значення активних S-блоків наведено разом, оскільки, як тільки різниці зафіксовано, значення можуть бути отримані з DDT (таблиці розподілу різниць) і навпаки.

1.6 Хронологія та класифікація атак на Grøstl та «Купини»

Аналіз стійкості геш-функцій з AES-подібними перетвореннями пройшов значний шлях від перших спроб застосування rebound-атаки до складних комбінованих методів. Нижче наведено хронологічний огляд ключових результатів для функцій Grøstl та Купуна, отриманих на основі аналізу джерел.

Аналіз стійкості Grøstl

Після публікації Grøstl як кандидата на конкурс SHA-3 основна увага криптоаналітиків була зосереджена на rebound-атаках. Окрім базових робіт Менделя (Mendel et al.), у 2010 році Томас Пейрін (Thomas Peyrin) у роботі [14] представив покращені диференціальні атаки. Використовуючи нові техніки побудови диференціальних шляхів, йому вдалося покращити складність атак на редуковані версії Grøstl та ЕШО, що продемонструвало важливість врахування ступенів свободи не лише у

вхідній фазі rebound-атаки, а й на етапах поширення різниць. Цей етап можна вважати новим кроком щодо стійкості, адже усі попередні аналізи Грьостла вивчали диференційну поведінку перестановок класичним способом. Тобто вони виводили диференціальні траєкторії, обробляючи два різні вхідні значення для кожної з перестановок P і Q (дві перестановки атакувалися окремо).

Ці перестановки імітували блоковий шифр AES. Внутрішня диференційна атака може застосовуватися, коли функція побудована на паралельних обчислювальних гілках, які не є достатньо відмінними. Хитрість полягає в тому, щоб розробити диференціальний шлях, що відображає відмінності між гілками, а не між двома вхідними даними функції. Зазвичай цього уникають, застосовуючи сильне розділення між двома паралельними гілками.

У випадку Grøstl розділення між перестановками P і Q є незначним, і тому можна використовувати цю властивість, щоб, наприклад, здійснити диференційну атаку на повну функцію стиснення Grøstl-256 [14].

Метою є знаходження двох внутрішніх станів A та B , таких, що їх вхідна різниця $A \oplus B = \Delta_{IN}$, а вихідна різниця задовольняє умову $P(A) \oplus Q(B) = \Delta_{OUT}$.

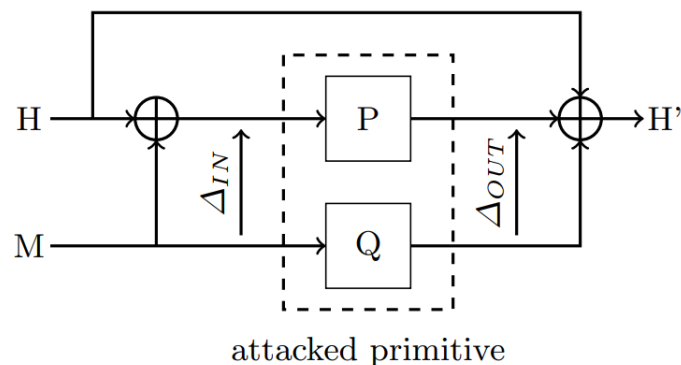


Рисунок 1.7 – Диференціальна атака на Grøstl [1]

Ефективність цього методу зумовлена архітектурною подібністю P та Q , які відрізняються лише етапом додавання констант

(AddRoundConstant), що дозволяє утримувати кількість активних різниць на низькому рівні. Для реалізації атаки використовуються усічені диференціали та техніка **Super-Sbox**.

Super-Sbox — це техніка криптоаналізу, що дозволяє розглядати два послідовні раунди AES-подібного шифру як один шар великих віртуальних S-boxів. Якщо раундова функція має структуру $SB \rightarrow SR \rightarrow MC \rightarrow ARK$, то об'єднання двох раундів дозволяє представити перетворення між шаром SB раунду i та шаром SB раунду $i + 1$ як набір незалежних 32-бітних (або 64-бітних) S-boxів. Це значно спрощує пошук диференціальних характеристик, дозволяючи аналітику «перестрибнути» через складний етап перемішування (MixColumns) з меншими обчислювальними витратами.

Використання Super-Sbox дозволяє зосередити обчислювально складну частину rebound-атаки у середніх раундах, залишаючи легші частини на початку та в кінці шляху. Такий підхід дозволив побудувати диференціальний шлях для 10 раундів функції стиснення Grøstl-256 [14].

Оцінки атак розрізнення (distinguishers), які були проведені у статті, наведено у таблиці 1.3.

Таблиця 1.3 – Зведені дані атак розрізнення на функцію стиснення Grøstl (адаптовано з [14])

Ціль атаки	Раунди	Обчислювальна складність	Вимоги до пам'яті	Тип атаки	Джерело
Функція стиснення Grøstl-256	7	2^{56}	—	розрізнявач	[9]
	8	2^{112}	2^{64}	розрізнявач	[6]
	9	2^{80}	2^{64}	розрізнявач	[14]
	10	2^{192}	2^{64}	розрізнявач	[14]
Функція стиснення Grøstl-512	11	2^{640}	2^{64}	розрізнявач	[14]

У 2011 році Мартін Шлеффер (Martin Schl  ffer) опублікував оновлений диференціальний аналіз [16], який уточнив оцінки стійкості Grøstl проти колізійних атак. Ця робота підтвердила, що, хоча функція

залишається стійкою при повній кількості раундів, запас міцності для варіантів зі зменшеною кількістю раундів потребує ретельного вивчення.

Окрім атак розрізнення, критично важливими є атаки на пошук колізій та прообразів. Це зумовлено тим, що саме ці класи атак визначають безпеку геш-функції у реальних криптографічних протоколах на прикладному рівні. Зокрема, успішне знаходження колізії повністю руйнує інфраструктуру відкритих ключів та електронного цифрового підпису, дозволяючи зловмиснику підробляти юридично значущі документи. Водночас атака на прообраз компрометує механізми автентифікації, системи безпечного зберігання паролів та перевірки цілісності даних. Отже, якщо успішна атака розрізнення є свідченням зламу криптографічної стійкості, то успішний пошук колізій або прообразів означає повний злам криптосистеми. Найкращі відомі результати наведено у таблиці 1.4.

Таблиця 1.4 – Найкращі відомі атаки на колізії та прообрази Grøstl

Функція	Тип атаки	Раунди	Складність	Джерело
Grøstl-256	Колізія	3 / 10	2^{64}	[16]
	Semi-free-start колізія	6 / 10	2^{112}	[9]
	Псевдо-прообраз	6 / 10	$2^{244.8}$	[15]
Grøstl-512	Колізія	3 / 14	2^{192}	[16]
	Semi-free-start колізія	7 / 14	2^{152}	[10]
	Псевдо-прообраз	8 / 14	$2^{507.2}$	[15]

Значним кроком вперед стала робота 2017 року авторства Кошоа, Гомес та Лерсі [1]. Автори запропонували новий вид атаки розрізнення (distinguishing attack) на AES-подібні перестановки. Їм вдалося побудувати розрізнявач для 11 раундів перестановки Grøstl-512, що на той момент було найкращим результатом. Цей результат базувався на вдосконаленні техніки rebound шляхом пошуку розв'язків для середніх

раундів з меншою складністю. Проведення наведених атак нижче є основою цього дослідження, адже тут включаються головні атаки, саме rebound-атаки на Grøstl, що вкотре підтверджує актуальність роботи. З плином часу помітно, як інтенсивно розвивається техніка даних атак на геш-функції, що використовують AES-подібні перетворення. У Таблиці 1.5 наведено результати rebound-атак:

Таблиця 1.5 – Rebound-атаки на внутрішні перестановки Grøstl-512

Раунди	Часова складність	Загальна атака	Пам'ять	Джерело
7	2^{152}	2^{512}	2^{56}	[1]
8	2^{280}	2^{448}	2^{64}	[1]
9	2^{328}	2^{384}	2^{64}	[1]
10	2^{392}	2^{448}	2^{64}	[1]
11	2^{72}	2^{96}	2^{256}	[1]

Алгоритм дозволяє знайти шукану пару зі складністю обчислень 2^{72} та вимогами до пам'яті 2^{56} . Для порівняння, загальна атака для таких вхідних та вихідних патернів вимагала б 2^{96} операцій.

Аналіз стійкості Курупа

Одразу після прийняття «Купини» китайські дослідники Цзянь Цзоу та Ле Донг опублікували один із перших незалежних аналізів нової функції [17]. Вони зосередилися на стійкості функції стиснення Курупа-256 до усіченого диференціального криптоаналізу (англ. truncated differential cryptanalysis). Дослідникам вдалося побудувати диференціальні шляхи для 4-х раундів функції стиснення з імовірністю, вищою за випадковий підбір (див. табл. 1.6).

Пізніше команда розробників та дослідників Grøstl (Мендель, Добрауніг, Айхльзедер) провела більш ґрунтовний аналіз «Купини», застосовуючи вдосконалені методи rebound-атак [3]. Вони показали, що

Таблиця 1.6 – Зведені результати атак на Курупа (адаптовано з [17])

Алгоритм	Ціль атаки	Тип атаки	Раунди	Час	Пам'ять	Джерело
Курупа-256	Геш-функція	Колізія	5	2^{120}	2^{64}	[17]
	Вихідне перетворення	Колізія	5	2^{240}	2^{64}	[17]
	Геш-функція	Псевдо-прообраз	6	$2^{250.33}$	$2^{250.33}$	[17]
Курупа-512	Вихідне перетворення	Прообраз	8	2^{472}	2^{120}	[17]
	Геш-функція	Псевдо-прообраз	8	$2^{498.33}$	$2^{498.33}$	[17]

через схожість S-boxів «Купини» та AES до неї застосовні ті ж методи побудови шляхів, що дозволило покращити результати атак на пошук колізій (Таблиця 1.7).

Таблиця 1.7 – Покращені атаки на колізії та прообрази Курупа [3]

Варіант	Тип атаки	Раунди	Складність	Автори
Курупа-256	Колізія	4 / 10	2^{67}	[3]
	Semi-free-start колізія	6 / 10	2^{52}	[3]
Курупа-512	Колізія	5 / 14	2^{160}	[3]
	Semi-free-start колізія	7 / 14	2^{139}	[3]

Хоча ця атака не загрожувала повній версії «Купини» (яка має 10 раундів), робота [17] та подальші дослідження стали важливою віхою, оскільки продемонстрували, що архітектурна близькість до Grøstl робить Купину вразливою до тих самих класів атак, і підтвердили коректність вибору запасу кількості раундів українськими розробниками.

2 ПОБУДОВА ТРИАНГУЛЯЦІЙНОЇ АТАКИ НА ГЕШ-ФУНКЦІЮ «КУПИНА»

Важливі уточнення методу побудови атаки

Як було описано в 1.5, під загальним кутом зору процес об'єднання кількох вхідних фаз для формування Super-Inbound можна розглядати як задачу розв'язання складної системи нелінійних рівнянь. У цій системі всі доступні ступені свободи (DoF) виступають у ролі змінних. Для того щоб таку систему можна було ефективно розв'язати за допомогою алгоритму триангуляції, вона повинна задовольняти дві фундаментальні умови:

I. Система не повинна бути перевизначеною. Це означає, що вона повинна мати розв'язки, а отже, загальна кількість доступних ступенів свободи має бути більшою за кількість рівнянь у системі. У випадку AES-подібних геш-функцій ідеальною вважається ситуація, коли після фіксації активних байтів залишається хоча б один «вільний байт», оскільки відсутність вільних змінних з високою ймовірністю призводить до нерозв'язності системи.

II. Ефективність розв'язання. Процес повинен зводитися до послідовного використання ступенів свободи шляхом фіксації певних змінних. Загальна обчислювальна складність атаки визначається кількістю тих змінних, які алгоритм триангуляції не зміг зафіксувати аналітично і які криптоаналітику доводиться знаходити методом повного перебору (brute-force).

Для практичної реалізації атаки на колізії (наприклад, для n -бітної геш-функції), загальні кроки для ідентифікації життєздатного диференціального шляху (trail) описуються наступним евристичним алгоритмом:

- 1) Знайти усічений диференціальний шлях, використовуючи моделі пошуку на основі обмеженого програмування;

2) Обрати кілька послідовних раундів (зазвичай від 2 до 4) як фазу Super-Inbound, що включає декілька локальних Inbound-частин. На цьому етапі фіксуються всі вхідні/вихідні різниці активних S-блоків, а відповідно, і їхні значення;

3) Побудувати систему рівнянь, яка з'єднає Inbound-частини (зокрема через розклад ключів або додавання констант). Перевірити, чи є ця система розв'язною за допомогою алгоритму триангуляції.

2.1 Аналіз застосовності триангуляційних атак до алгоритму «Купина»

Постановка задачі для подальшого дослідження.

Описаний теоретичний апарат триангуляційної атаки та евристичний метод були успішно застосовані для зламу редукованих версій геш-функції Grøstl. Проте критичною точкою цього методу є Крок 3 — здатність побудувати розв'язну систему алгебраїчних рівнянь. У наступних підрозділах буде наведено власне практичне дослідження того, як заміна лінійної операції XOR ($\oplus$) на модульне арифметичне додавання ($\boxplus$) в архітектурі ДСТУ 7564:2014 «Купина» впливає на виконання Умови I та Умови II, та чи можливо здійснити Крок 3 евристичного алгоритму для перестановки T^+ . Для оцінки стійкості національного стандарту гешування до найсучасніших методів криптоаналізу застосуємо алгоритм триангуляційної Rebound-атаки до архітектури алгоритму «Купина». Процес перенесення атаки розділимо на аналіз двох незалежних перестановок: $T^\oplus$ та T^+ .

Адаптація Inbound-фази для гілки $T^\oplus$

Перестановка $T^\oplus$ використовує операцію побітового додавання за модулем 2 (XOR) для ін'єкції раундових констант. Оскільки всі операції цієї гілки виконуються виключно над полем Галуа $GF(2^8)$, алгебраїчні властивості диференціальних шляхів зберігаються.

Адаптація атаки вимагає обчислення нових таблиць розподілу різниць (DDT) для чотирьох нелінійних замінів $(\pi_0, \pi_1, \pi_2, \pi_3)$. Оскільки лінійне перетворення спирається на MDS-матрицю, математичний апарат побудови «станів-трикутників» (triangulating states) успішно переноситься на перестановку $T^\oplus$. Теоретично, зловмисник здатний розв'язати систему лінійних рівнянь і побудувати детерміновану Inbound-фазу.

Злам диференціального шляху в гілці T^+

Критична точка перенесення атаки виникає під час спроби застосувати ті самі диференціальні характеристики до перестановки T^+ . На відміну від $T^\oplus$, ін'єкція констант тут виконується операцією додавання за модулем 2^{64} ($\boxplus$).

Розглянемо математичну модель руйнування диференціалу під час цієї операції. Нехай стан однієї колонки матриці представлено як 64-бітне слово x , яке можна розбити на байти:

$$x = (x_7, x_6, \dots, x_1, x_0), \quad \text{де } x_i \in \mathbb{Z}_{256} \quad (2.1)$$

Аналогічно подамо раундову константу C :

$$C = (c_7, c_6, \dots, c_1, c_0) \quad (2.2)$$

Розрахунок оригінального вихідного стану y та стану зі збуренням y' відбувається за модулем 2^{64} :

$$\begin{aligned} y &= f(x) = (x + C) \pmod{2^{64}} \\ y' &= f(x \oplus \alpha) = ((x \oplus \alpha) + C) \pmod{2^{64}} \end{aligned} \quad (2.3)$$

Припустимо, що в Inbound-фазі ми внесли мінімальний диференціал α , локалізований виключно в наймолодшому байті:

$$\alpha = (0, 0, \dots, 0, \alpha_0) \quad (2.4)$$

У лінійній XOR-схемі вихідна різниця $y \oplus y'$ дорівнювала б α . Проте

під час арифметичного додавання вихідний вектор $y = (y_7, y_6, \dots, y_0)$ формується з урахуванням каскадного переносу бітів між байтами. Значення кожного i -го байта обчислюється як:

$$y_i = (x_i + c_i + \gamma_i) \pmod{256} \quad (2.5)$$

де γ_i — біт переносу (carry bit) з попереднього, молодшого байта, який розраховується за формулою:

$$\gamma_i = \left\lfloor \frac{x_{i-1} + c_{i-1} + \gamma_{i-1}}{256} \right\rfloor, \quad \text{причому } \gamma_0 = 0 \quad (2.6)$$

Саме генерація переносів γ_i перетворює поширення різниці на ймовірнісний процес. У стані y' значення переносів будуть іншими через наявність α_0 . Як наслідок, вихідний вектор різниці розсіюється на старші розряди і набуває вигляду:

$$\Delta y = y \oplus y' = (0, \dots, 0, \underbrace{*, *, \dots, *}_t) \quad (2.7)$$

де t — кількість байтів, що змінили своє значення ($t \in \{1, 2, \dots, 8\}$).

Спроба криптоаналітика замінити оператор $\oplus$ на арифметичне додавання $+$ («без переносів») для спрощення системи рівнянь є неможливою, оскільки це суперечить базовій алгебраїчній структурі алгоритму. Таким чином, ймовірність збереження локалізованої різниці $P(t = 1)$ експоненційно падає з кожним раундом. Ступені свободи атакуючого миттєво вичерпуються на спробах обчислити систему рівнянь з невідомими нелінійними змінними γ_i .

Аналіз диференціальних перетворень на рівні бітових векторів

Для глибшого розуміння механізму руйнування диференціального шляху в Кируна розглянемо процес виникнення нелінійних переносів на прикладі конкретних бітових послідовностей. Нехай ми маємо редукований 16-бітний стан (2 байти). Проаналізуємо, як вхідна зміна в

один біт розповсюджується під час операції $\boxplus$.

Нехай вхідні параметри системи мають такі шістнадцяткові значення:

– Вхідний стан: $x = 0x12A4$ (у двійковому вигляді: 00010010 10100100);

– Раундова константа: $c = 0x56B3$ (у двійковому вигляді: 01010110 10110011);

– Вхідна різниця: $\alpha = 0x0001$ (зміна молодшого біта).

Обчислимо оригінальний результат додавання $y = (x + c) \pmod{2^{16}}$:

$$\begin{array}{r} 00010010 \ 10100100 \ (x) \\ + 01010110 \ 10110011 \ (c) \\ \hline 01101001 \ 01010111 \ (y = 0x6957) \end{array}$$

Тепер обчислимо стан зі збуренням $x' = x \oplus \alpha = 0x12A5$ та результат $y' = (x' + c) \pmod{2^{16}}$:

$$\begin{array}{r} 00010010 \ 10100101 \ (x') \\ + 01010110 \ 10110011 \ (c) \\ \hline 01101001 \ 01011000 \ (y' = 0x6958) \end{array}$$

Знайдемо вихідну різницю $\Delta y = y \oplus y'$:

$$\Delta y = 0x6957 \oplus 0x6958 = 0x002F \quad (2.8)$$

В аналізованому прикладі вихідна різниця Δy становить 00000000 00101111. Це демонструє, що початкова різниця в 1 біт призвела до зміни 5 бітів у молодшому байті. Проте оскільки в даному випадку каскад переносів γ_i зупинився на 6-му біті, старший байт залишився незмінним ($t = 1$).

Для того, щоб відбулося розповсюдження різниці на старший розряд ($t = 2$), необхідно, щоб ланцюжок переносів пройшов через усі біти молодшого байта. Ймовірність такої події для рівномірно розподілених

випадкових значень x та s становить 2^{-8} . Таке поширення переносу підтверджує, що операція додавання в Кируна діє як нелінійний розсіювач диференціалів, складність аналізу якого зростає експоненційно з кожним наступним байтом у 64-бітному слові.

Оцінка статистичної збіжності експерименту

Оскільки поширення переносу в операції модульного додавання є ймовірнісним процесом, точність отриманих результатів залежить від розміру вибірки. Згідно з теоретичною моделлю, ймовірність того, що каскад переносів пройде через усі 8 бітів молодшого байта і змінить старший байт, становить $P_{theor} = 2^{-8} \approx 0.003906$ (або 0.39%).

Для підтвердження теоретичної моделі було проведено серію експериментів із поступовим збільшенням кількості ітерацій N . Результати зведено у таблицю 2.1.

Таблиця 2.1 – Залежність емпіричної ймовірності руйнування диференціалу від кількості ітерацій

Кількість ітерацій (N)	Кількість переносів ($t = 2$)	Емпірична ймовірність
1 000	1	0.10
10 000	36	0.36
100 000	422	0.42
1 000 000*	$\approx 3\,906$	≈ 0.39

Результати під час проведення дослідження для 2-х байтових векторів

Аналізуючи дані з таблиці 2.1, можна спостерігати дію закону великих чисел: при малій вибірці ($N = 1000$) статистична похибка є значною, проте зі збільшенням кількості ітерацій до 100 000 емпірична ймовірність збігається з розрахунковим теоретичним значенням у 0.39%. Це повністю підтверджує коректність програмної моделі та математично доводить імовірнісну природу руйнування диференціалів у гілці T^+ . Програмна реалізація знаходиться в публічному репозиторії за цим [посиланням](#).

Аналіз впливу нелінійних перетворень та побудова таблиць розподілу різниць

Як зазначалося раніше, перестановка $T^\oplus$ використовує виключно побітове додавання за модулем 2, зберігаючи класичні властивості розповсюдження диференціалів. Проте критичною відмінністю архітектури Куруна від фіналіста SHA-3 Grøstl є шар нелінійного бієктивного відображення (SubBytes).

Алгоритм Grøstl використовує єдиний стандартний S-блок, запозичений з AES. Для реалізації алгоритму триангуляції та проходження Inbound-фази криптоаналітику достатньо обчислити лише одну таблицю розподілу різниць (Difference Distribution Table — DDT). Ця таблиця показує, скільки існує таких вхідних значень x , для яких різниця на вході α призведе до різниці β на виході S-блоку. Елементи цієї таблиці визначаються за такою формулою:

$$DDT(\alpha, \beta) = \left| \{x \in GF(2^8) \mid S(x) \oplus S(x \oplus \alpha) = \beta\} \right| \quad (2.9)$$

де:

- α — різниця байтів на вході до S-блоку;
- β — різниця байтів на виході з S-блоку;
- $S(x)$ — функція нелінійного перетворення (власне S-блок);
- $|\{\dots\}|$ — кількість елементів у множині (тобто скільки разів виконується умова).

Натомість стандарт Куруна застосовує чотири різні нелінійні підстановки ($\pi_0, \pi_1, \pi_2, \pi_3$). Вибір конкретного S-блоку для байта матриці стану залежить від його індексу (позиції). Це означає, що для побудови системи алгебраїчних рівнянь під час Кроку **2**) алгоритму триангуляції, злоумисник змушений:

- 1) Згенерувати чотири незалежні таблиці DDT розміром 256×256 кожна.
- 2) Під час розрахунків динамічно визначати індекс кожного

активного байта в Super-Inbound фазі, щоб зрозуміти, до якої з чотирьох таблиць потрібно звертатися.

Хоча наявність чотирьох S-блоків не робить систему нерозв'язною, вона штучно порушує симетрію перестановки. У класичній атаці Донга на Grøstl симетрія одного єдиного S-блоку дозволяла групувати змінні та суттєво оптимізувати процес триангуляції. У випадку «Купини» алгебраїчна неоднорідність шару SubBytes значно ускладнює програмну реалізацію пошуку вільних змінних g_{free} . Це підвищує комбінаторну складність побудови розрізнявача ще на етапі ініціалізації атаки.

Оцінка обчислювальної складності триангуляційної атаки

В алгебраїчному криптоаналізі обчислювальна складність C визначається кількістю змінних, які не вдалося зафіксувати за допомогою алгоритму розв'язання рівнянь (наприклад, триангуляції), і які потребують подальшого повного перебору (brute-force).

Нехай g — кількість невідомих байтів стану, які криптоаналітик повинен підібрати випадковим чином для задоволення системи рівнянь Inbound-фази. Тоді часова складність етапу складатиме:

$$C_{inbound} \approx 2^{8 \cdot g} \quad (2.10)$$

Для успішного застосування атаки на пошук колізій необхідно, щоб загальна складність алгоритму C_{attack} була строго меншою за криптографічну межу стійкості, визначену парадоксом днів народжень для n -бітної геш-функції:

$$C_{attack} < 2^{n/2} \quad (2.11)$$

Для стандарту Кірупа-256 ця межа становить 2^{128} операцій.

Як було доведено під час програмного моделювання, використання модульного арифметичного додавання $\boxplus_{64}$ генерує каскад невідомих бітів переносу γ_i . У кожній 64-бітній колонці матриці стану (яка складається з 8 байтів) ця операція генерує до 7 невідомих міжбайтових переносів. У матриці Кірупа-256 використовується 8 таких колонок. Отже, всього за

один раунд перестановки T^+ генерується до $7 \times 8 = 56$ прихованих нелінійних булевих змінних, які неможливо виразити через лінійні рівняння алгоритму триангуляції.

Спроба інтегрувати ці змінні у систему рівнянь призводить до того, що кількість невідомих стрімко перевищує кількість наявних ступенів свободи. Алгоритм триангуляції не здатен звести таку систему до розв'язного вигляду.

Як наслідок, єдиним методом проходження Inbound-фази для гілки T^+ залишається імовірнісне вгадування значень переносів та відповідних байтів стану. Для того, щоб переноси задовольнили вимоги диференціального шляху (утримати різницю локалізованою), криптоаналітик повинен контролювати майже весь 512-бітний стан. Кількість байтів, які необхідно перебрати (g), наближається до повного розміру стану (64 байтів). Підставивши це значення у формулу складності, отримуємо:

$$C_{attack} \approx 2^{8 \cdot 64} = 2^{512} \quad (2.12)$$

Оцінка стійкості повної функції стиснення

Функція стиснення Куруна формується як об'єднання двох гілок:

$$f(H, M) = T^{\oplus}(H \oplus M) \oplus T^+(M) \oplus H \quad (2.13)$$

Щоб знайти колізію для повної функції, криптоаналітик повинен узгодити вихідні диференціали обох перестановок. Оскільки триангуляційна Rebound-атака детерміновано працює лише для гілки $T^{\oplus}$, гілка T^+ діє як ідеальний нелінійний ізолятор. Будь-який структурований диференціальний слід, згенерований у $T^{\oplus}$, нівелюється нелінійним розсіюванням t -вектора (каскадом переносів).

Теоретичне моделювання із застосуванням побайтового аналізу переносів доводить, що архітектура Куруна має структурний імунітет до триангуляційних атак. Наявність модульного додавання руйнує цілісність

алгебраїчного кільця, роблячи побудову розрізнявача для повної функції стиснення обчислювально неможливою. Отримана обчислювальна складність (2^{512}) значно перевищує межу стійкості до пошуку колізій (2^{128}), через що атака деградує і стає суттєво менш ефективною, ніж звичайний метод повного перебору. Це математично підтверджує абсолютну практичну неможливість реалізації даного класу атак на стандарт Кірупа.

2.2 Формальна модель перенесення триангуляційної атаки на модифіковану Купину

Для точного доведення того, що зростання обчислювальної складності атаки зумовлене виключно архітектурою додавання за модулем 2^{64} , вводиться модифікована перестановка T^{*+} . У цій перестановці раундове додавання констант AC виконується не над 64-бітними словами, а над ізольованими байтами в кільці $\mathbb{Z}_{256}$.

Нехай внутрішній стан на вході до операції AC подано як матрицю $X = (x_{i,j})$, а раундову константу як $C = (c_{i,j})$, де $0 \leq i, j \leq 7$, $x_{i,j}, c_{i,j} \in \mathbb{Z}_{256}$. Тоді модифікована операція AC^* визначається як:

$$y_{i,j} = (x_{i,j} + c_{i,j}) \pmod{2^8} \quad (2.14)$$

Розглянемо поширення диференціалу в цій операції. Нехай $\Delta X = X \oplus X'$ — вхідна різниця (XOR-диференціал). Для будь-якого байта $x_{i,j}$, який має нульову різницю ($\Delta x_{i,j} = 0$, тобто $x_{i,j} = x'_{i,j}$), вихідна різниця розраховується детерміновано:

$$\begin{aligned} \Delta y_{i,j} &= y_{i,j} \oplus y'_{i,j} = ((x_{i,j} + c_{i,j}) \pmod{2^8}) \oplus ((x'_{i,j} + c_{i,j}) \pmod{2^8}) \\ &= ((x_{i,j} + c_{i,j}) \pmod{2^8}) \oplus ((x_{i,j} + c_{i,j}) \pmod{2^8}) = 0 \end{aligned} \quad (2.15)$$

Оскільки міжбайтові переноси γ штучно усунуті, умова

$\Delta x_{i,j} = 0 \implies \Delta y_{i,j} = 0$ виконується з імовірністю $P = 1$. Це алгебраїчно ізолює кожен байт матриці стану, відновлюючи властивість побітової незалежності, притаманну алгоритму Grøstl, а отже, на даному етапі до «Купини» може бути застосований повний алгоритм триангуляційної атаки Донга [4].

Алгоритм атаки та розрахунок обчислювальної складності

Нехай n — розмір геш-коду ($n = 256$), S_{in} — кількість активних S-блоків у вхідній фазі (Super-Inbound), g — кількість «вільних змінних» (байтів), які неможливо обчислити аналітично і які вимагають повного перебору. Алгоритм атаки складається з наступних кроків:

1) **Побудова системи рівнянь.** Використовуючи властивість детермінованого поширення нульових різниць (рівняння (2.15)), формується розв’язна система алгебраїчних рівнянь для активних S-блоків.

2) **Триангуляція.** Алгоритм триангуляції зводить систему рівнянь до трикутного вигляду, послідовно фіксуючи змінні. Кількість ступенів свободи (DoF) 64-байтового стану достатня для задоволення системи, залишаючи g_{free} невідомих байтів.

3) **Обчислення Inbound-фази.** Для всіх можливих $2^{8 \cdot g_{free}}$ значень вільних байтів розраховується відповідний диференціальний слід Super-Inbound фази. Складність цього етапу становить $2^{8 \cdot g_{free}}$ операцій.

4) **Outbound-фаза.** Отримані значення поширюються у прямому та зворотному напрямках (Outbound-фаза) для досягнення колізії на виході функції стиснення. Ймовірність успішного проходження усічених диференціалів становить $P_{out} = 2^{-p}$, де p — кількість умов фільтрації.

5) **Повторення.** Кроки 1-4 повторюються 2^p разів для знаходження повної колізії.

Загальна складність атаки C_{mod} розраховується за формулою:

$$C_{mod} = 2^p \cdot 2^{8 \cdot g_{free}} \quad (2.16)$$

Згідно з результатами застосування триангуляційної атаки до 512-бітних станів (наприклад, перестановка Grøstl-512 [4]), кількість вільних байтів після триангуляції становить $g_{free} \approx 9$. Враховуючи ймовірність Outbound-фази, загальна часова складність складатиме:

$$C_{mod} \approx 2^{8 \cdot 9} = 2^{72} \text{ операцій обчислення функції} \quad (2.17)$$

Для модифікованої перестановки «Купина»-512 ($n = 512$, розмір стану 1024 біти) розмір матриці збільшується вдвічі. Відповідно, кількість вільних змінних g_{free} зростає пропорційно і становить ≈ 21 байт. Тоді складність атаки складатиме:

$$C_{mod(512)} \approx 2^{8 \cdot 21} = 2^{168} \text{ операцій} \quad (2.18)$$

Алгебраїчна формалізація та оцінка просторової складності атаки

Процес триангуляційної атаки на Курупа можна представити як пошук розв'язків системи булевих рівнянь. Для 256-бітної версії Курупа внутрішній стан складається з 64 байтів, кожен з яких представляється вектором з 8 змінних.

Повна система рівнянь для одного раунду перестановки T^+ містить:

- Рівняння нелінійного шару (S-блоки): $y = \pi(x)$, де кожний S-блок описується системою нелінійних рівнянь ступеня $d \leq 7$;
- Рівняння лінійного шару (ψ): лінійні комбінації над $GF(2^8)$;
- Рівняння арифметичного додавання ($\boxplus$) включають змінні переносів γ_i , які суттєво підвищують алгебраїчний ступінь системи.

Окрім часової складності, критичним параметром є просторова складність — обсяг пам'яті, необхідний для зберігання проміжних результатів Inbound-фази. Для успішної побудови таблиці відповідності (як показано на алгоритмі Донга) необхідно зберегти всі згенеровані розв'язки Inbound-фази. Кількість цих базових станів безпосередньо визначається кількістю доступних вільних байтів g_{free} і становить $2^{8 \cdot g_{free}}$.

Для модифікованої Курупа-256 (без переносів), де кількість вільних байтів становить $g_{free} \approx 9$, обсяг необхідної пам'яті M визначається як:

$$M = 2^{8 \cdot 9} \times 64 \text{ байт} \approx 2^{72} \times 2^6 = 2^{78} \text{ байт} \quad (2.19)$$

Переводячи в загальноприйняті одиниці виміру:

$$M \approx \frac{2^{78}}{2^{40}} \text{ ТБ} \approx 2^{38} \text{ ТБ} \quad (2.20)$$

Такий обсяг пам'яті (а це мільярди терабайт) є фізично недосяжним для сучасних обчислювальних систем. Для оригінальної архітектури Курупа через наявність переносів кількість вільних змінних відсутня, що теоретично вимагає зберігання 2^{512} станів. Це доводить, що Курупа має не лише часовий, але й абсолютний просторовий імунітет до даного класу алгебраїчних атак.

Порівняльний аналіз складностей

Отримані результати дозволяють провести строгу оцінку криптографічної міцності оригінальної та модифікованої архітектур:

– **Модифікована Курупа-256:** $C_{mod} = 2^{72} \ll 2^{128}$ (Межа стійкості). Атака успішно знаходить колізію; обчислювальна складність є меншою за повний перебір.

– **Модифікована Курупа-512:** $C_{mod(512)} = 2^{168} \ll 2^{256}$. Атака успішна.

Водночас повернення до оригінальної архітектури T^+ (з генерацією $7 \times 8 = 56$ нелінійних переносів γ_i у кожному раунді) перетворює систему рівнянь на перевизначену. У цьому випадку алгоритм триангуляції на кроці 2) не знаходить аналітичних розв'язків, і кількість вільних байтів g_{free} стає рівною повному об'єму стану (64 байти).

Тоді складність атаки на оригінальний стандарт ДСТУ 7564:2014 становить:

$$C_{attack} = 2^{8 \cdot 64} = 2^{512} \gg 2^{128} \quad (2.21)$$

Для наочного відображення впливу модифікації на криптографічну стійкість зведемо розраховані теоретичні показники до таблиці 2.2. За базовий еталон взято результати застосування атаки на перестановку Grøstl-512 [4].

Отримані результати дозволяють провести строгу оцінку криптографічної міцності оригінальної та модифікованої архітектур. Максимальна кількість раундів, на яку може бути поширена атака, жорстко обмежується стартовим бюджетом вільних змінних (C_{mod}) та диференціальним числом розгалуження лінійного шару.

Під час Outbound-фази поширення різниці має ймовірнісний характер. Оскільки раундова функція містить два послідовні лінійні перетворення — циклічний зсув та множення на MDS-матрицю, вони сумарно забезпечують сильний лавинний ефект, стрімко збільшуючи кількість активних байтів у стані. Щоб утримати усічений диференціальний шлях вузьким і не дозволити різниці заповнити всю матрицю стану, криптоаналітик змушений покладатися на ймовірнісне звуження диференціалу під час проходження шару ψ (наприклад, з 8 активних байтів до 1). Оскільки матриця ψ діє над $GF(2^8)$, ймовірність того, що лінійна комбінація перетворить 7 активних байтів на нульові для одного стовпця, становить $P_{8 \rightarrow 1} = (2^{-8})^7 = 2^{-56}$. Кожен такий перехід коштує 2^{56} стартових пар і дозволяє подовжити диференціальний шлях на два раунди (один ймовірнісний раунд звуження $8 \rightarrow 1$ і один наступний детермінований раунд розширення $1 \rightarrow 8$).

Виходячи з цього, гранична кількість раундів розраховується наступним чином:

- **Для 256-бітної архітектури:** Кількість бітів у 2^{72} пар вистачає лише на проходження одного звуження $8 \rightarrow 1$ ($2^{72} \times 2^{-56} = 2^{16}$). Це дозволяє додати 2 контрольовані раунди та 1 фінальний неконтрольований раунд розширення на краю шляху. Сумарна кількість раундів: 4 (Inbound) + 3 (Outbound) = **7 раундів**.

- **Для 512-бітної архітектури:** 2^{168} пар дозволяє пройти рівно

три ймовірнісні звуження $8 \rightarrow 1$ (сумарна ймовірність 2^{-168}). Це дає змогу розтягнути шлях на 6 контрольованих раундів та 1 фінальний неконтрольований. Сумарна кількість раундів: 4 (Inbound) + 7 (Outbound) = **11 раундів**.

Таблиця 2.2 – Оцінка складності триангуляційної атаки на пошук колізій для оригінальної та модифікованої архітектур

Алгоритм	Ціль атаки	Тип атаки	Раунди	Час	Джерело
Grøstl-256	Функція стиснення	Колізія	7 / 10	2^{72}	[4]
Grøstl-512	Функція стиснення	Колізія	11 / 14	2^{168}	[4]
Модиф. Курупа-256	Функція стиснення	Колізія	7 / 10	2^{72}	Розділ 2.1
Модиф. Курупа-512	Функція стиснення	Колізія	11 / 14	2^{168}	Розділ 2.1

Водночас повернення до оригінальної архітектури T^+ (з генерацією нелінійних переносів γ_i) перетворює систему рівнянь на перевизначену. У цьому випадку алгоритм триангуляції не знаходить аналітичних розв'язків, і кількість вільних байтів g_{free} стає рівною повному об'єму стану (64 байти). Тоді складність атаки на оригінальний стандарт ДСТУ 7564:2014 катастрофічно зростає:

$$C_{attack} = 2^{8 \cdot 64} = 2^{512} \gg 2^{128} \quad (2.22)$$

Наведені формальні розрахунки математично доводять, що триангуляційна атака є високоефективною проти класичної AES-подібної архітектури без переносів (складність падає до 2^{72} та 2^{168} відповідно). Проте введення операції додавання за модулем 2^{64} руйнує детерміноване поширення нульових різниць, збільшуючи обчислювальну складність до 2^{512} операцій, що робить алгоритм «Купина» стійким до алгебраїчного криптоаналізу цього типу.

2.3 Аналіз лінійного перетворення та властивостей MDS-матриць

Окрім нелінійного шару SubBytes, на складність побудови диференціальних шляхів та розв’язання системи рівнянь алгоритмом триангуляції безпосередньо впливає лінійне перетворення (MixColumns, у ДСТУ 7564:2014 позначається як ψ). Дане перетворення забезпечує ефект дифузії у внутрішньому стані геш-функції.

В архітектурах Grøstl та «Купина» лінійне перетворення базується на використанні MDS-матриць (Maximum Distance Separable). Згідно з теорією кодування, лінійний код над полем Галуа $GF(2^8)$ довжиною n та розмірністю k має мінімальну відстань d , що обмежена межею Сінглтона: $d \leq n - k + 1$. MDS-матриця дозволяє досягти максимального значення цієї відстані, що в криптографії визначається як показник розгалуження (branch number) B .

Для матриці перемішування розміром 8×8 , що застосовується в аналізованих функціях, диференціальне число розгалуження становить $B = 9$. Це математично гарантує, що для будь-якого ненульового вхідного вектора X та відповідного вихідного вектора $Y = \psi(X)$, сума їхніх ненульових байтів (ваг Геммінга для векторів стану) задовольняє нерівність:

$$wt(X) + wt(Y) \geq 9 \quad (2.23)$$

Під час виконання Кроку 2) алгоритму триангуляції наявність MDS-матриці створює щільні лінійні залежності між байтами стану. Якщо на вході шару ψ відомий лише один активний байт різниці ($wt(X) = 1$), на виході гарантовано генерується 8 активних байтів ($wt(Y) = 8$). Ця властивість змушує криптоаналітика вводити значну кількість нових змінних для опису кожного вихідного байта, що пришвидшує вичерпання доступних ступенів свободи (DoF). Таким чином, характеристики MDS-матриці є визначальним фактором, який

обмежує здатність алгоритму триангуляції поширюватися на додаткові раунди без перевизначення системи рівнянь.

Програмна модель для емпіричного аналізу поширення диференціалів

Для верифікації розрахованих імовірностей руйнування диференціалів під дією модульного додавання $\boxplus_{64}$ була розроблена програмна релізація. Архітектура реалізує покрокове виконання операцій додавання констант до редукованого внутрішнього стану алгоритму.

Середовищем розробки обрано мову програмування Python. Програмна модель складається з трьох основних модулів: модуля генерації станів, модуля виконання арифметичних операцій та модуля статистичного аналізу.

Процес симуляції базується на такому алгоритмі:

1) **Формування векторів:** Генеруються базовий стан x та раундова константа c . Розмірність векторів для даного експерименту обмежена 16 бітами ($x, c \in [0, 2^{16} - 1]$) для ізоляції переходу переносу між двома суміжними байтами.

2) **Внесення різниці:** Формується збурений вектор $x' = x \oplus \alpha$, де $\alpha = 0x0001$.

3) **Модульне додавання:** Виконується обчислення функцій $y = (x + c) \pmod{2^{16}}$ та $y' = (x' + c) \pmod{2^{16}}$.

4) **Результат:** Обчислюється вихідна різниця $\Delta y = y \oplus y'$. Значення зсувається вправо на 8 бітів ($\Delta y \gg 8$). Якщо отримане значення дорівнює нулю, різниця локалізована в першому байті ($t = 1$). Якщо значення відмінне від нуля, фіксується розповсюдження різниці на старший байт через каскад переносів.

Набір статистичних даних формується шляхом збільшення вибірки N від 10^3 до 10^6 . Результати записуються у масиви для подальшого аналізу на збіжність до теоретичного значення 2^{-8} . Такий підхід мінімізує похибку вимірювання та забезпечує репрезентативність отриманих емпіричних даних (табл. 2.1).

ВИСНОВКИ

У роботі було проведено збір досліджень стосовно методів криптоаналізу Grøstl-подібних геш-функцій. Досліджено, що собою представляють Grøstl-подібні геш-функції, побачили їх особливості. Сформовано уявлення про таких яскравих представників геш-функцій з AES-перетворенням як **Grøstl** та **Kupyna**. Розглянуто їх побудову, особливості та відмінності в межах одного сімейства алгоритмів. Досліджено rebound-атаки, які є основним інструментом проти геш-функцій з AES-перетворення. Також детально розглянуто криптостійкість щодо цих та інших атак.

Незважаючи на час, що минув з моменту конкурсу SHA-3, інтерес спільноти до аналізу **Grøstl** зберігається, а методи атак удосконалюються. Зокрема, у відносно новій роботі 2017 року автори досліджують атаки розрізнення (distinguishing attacks) вже на 11-й раунд перестановки Grøstl-512. Цей результат є показовим у порівнянні зі станом досліджень попередніх років: так, ще у 2010 році стверджували, що на той момент атак на Grøstl-512 не було навіть проведено. Така динаміка свідчить про те, що Grøstl залишається важливим полігоном для тестування новітніх методів криптоаналізу структур з AES-перетворенням. Як було сказано раніше, Grøstl є фіналістом SHA-3 і за оглядом джерел фіналістом з чіткими доказами проти великого класу диференціальних атак. Дослідження показали, що внутрішні перестановки P і Q мають певні вади, що дозволяють будувати розрізнявачі (distinguishers) майже на повну кількість раундів (наприклад, 10 з 10 для Grøstl-256). Однак, завдяки конструкції з широким каналом (wide-pipe) та функції виходу, колізійні атаки залишаються ефективними лише для усічених версій (до 3-6 раундів), що зберігає практичну стійкість алгоритму.

Kupyna представляє собою український стандарт, який

продемонстрував схожу динаміку стійкості. Архітектурна спорідненість з Grøstl робить його вразливим до тих самих класів атак (зокрема, rebound). Проте також пропонуються атаки колізій та атаки прообразу на геш-функцію Купуна. Як було показано, Купуна схожий на Grøstl, але деякі константи раунду додаються з модульним додаванням. Такий підхід до структури ускладнює застосування багатьох відомих атак. Також, ми не можемо побудувати колізійну атаку на геш-функцію Купуна тільки за допомогою оригінальної rebound-атаки. Знайдені колізії для 4-5 раундів та псевдо-прообрази для 6-8 раундів підтверджують, що запас міцності (10 раундів для 256-бітної версії та 14 для 512-бітної) обрано коректно, з певною надлишковістю, адже на сьогоднішній день, в контексті класичної криптографії, не було проведено атак на всі раунди що **Купуна-256** так і **Купуна-512**. Використання чотирьох різних S-boxів у Купіні ускладнює деякі алгебраїчні атаки, проте не є панацеєю від диференціальних методів.

Незважаючи на теоретичні успіхи криптоаналітиків у побудові розрізнявачів, на сьогодні не існує практичних атак, які б загрожували цілісності повних версій Grøstl або «Купини». Обидві функції демонструють високий рівень надійності. Проте тенденція до вдосконалення rebound-атак вимагає постійного моніторингу запасу міцності цих алгоритмів.

Підсумовуючи, дослідження підтверджує, що геш-функції з AES-перетворенням, і зокрема **Купуна** та **Grøstl**, залишаються доволі **стійкими**. Але вони вимагають ретельного вибору кількості раундів, оскільки специфічні методи аналізу (Rebound, Super-Sbox) дозволяють ефективно ламати частину їх раундів, що може бути загрозою вразливості для них у майбутньому.

ПЕРЕЛІК ПОСИЛАНЬ

- [1] Victor Cauchois, Clément Gomez та Reynald Lercier. «Grøstl distinguishing attack: a new rebound attack of an AES-like permutation». B: *IACR Transactions on Symmetric Cryptology* (2017), с. 1—23.
- [2] Ivan Bjerre Damgård. «A design principle for hash functions». B: *Conference on the Theory and Application of Cryptology*. Springer. 1989, с. 416—427.
- [3] Christoph Dobraunig, Maria Eichlseder та Florian Mendel. «Analysis of the Kupyna-256 Hash Function». B: *Topics in Cryptology – CT-RSA 2016*. T. 9610. Lecture Notes in Computer Science. Available as Cryptology ePrint Archive, Report 2015/956. Springer, 2016, с. 165—181. DOI: 10.1007/978-3-319-29485-8_10.
- [4] Xiaoyang Dong та ін. «Triangulating rebound attack on AES-like hashing». B: *Annual International Cryptology Conference*. Springer. 2022, с. 94—124.
- [5] Praveen Gauravaram та ін. «Grøstl-a SHA-3 candidate». B: Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2009.
- [6] Henri Gilbert та Thomas Peyrin. «Super-Sbox cryptanalysis: improved attacks for AES-like permutations». B: *International Workshop on Fast Software Encryption*. Springer. 2010, с. 365—383.
- [7] Dmitry Khovratovich, Alex Biryukov та Ivica Nikolić. «Speeding Up Collision Search for Byte-Oriented Hash Functions». B: *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*. T. 5473. Lecture Notes in Computer Science. Springer, 2009, с. 164—181. DOI: 10.1007/978-3-642-00862-7_11.

- [8] Krystian Matusiewicz та ін. «Rebound attack on the full LANE compression function». B: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2009, c. 106—125.
- [9] Florian Mendel та ін. «Improved cryptanalysis of the reduced Grøstl compression function, ECHO permutation and AES block cipher». B: *International Workshop on Selected Areas in Cryptography*. Springer. 2009, c. 16—35.
- [10] Florian Mendel та ін. «Rebound attacks on the reduced Grøstl hash function». B: *Cryptographers' Track at the RSA Conference*. Springer. 2010, c. 350—365.
- [11] Florian Mendel та ін. «The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl». B: *International Workshop on Fast Software Encryption*. Springer. 2009, c. 260—276.
- [12] Ralph C Merkle. «One way hash functions and DES». B: *Conference on the Theory and Application of Cryptology*. Springer. 1989, c. 428—446.
- [13] Roman Oliynykov та ін. *A New Standard of Ukraine: The Kupyna Hash Function*. Cryptology ePrint Archive, Paper 2015/885. 2015. URL: <https://eprint.iacr.org/2015/885>.
- [14] Thomas Peyrin. «Improved differential attacks for ECHO and Grøstl». B: *Annual Cryptology Conference*. Springer. 2010, c. 370—392.
- [15] Yu Sasaki, Lei Wang та Kazumaro Aoki. «Pseudo-preimage Attacks on Grøstl and Others». B: *Selected Areas in Cryptography – SAC 2012*. T. 7707. Lecture Notes in Computer Science. Springer, 2013, c. 221—238. DOI: 10.1007/978-3-642-35999-6_15.
- [16] Martin Schl  ffer. «Updated differential analysis of Grøstl». B: *Gr  stl website (January 2011)* (2011).
- [17] Jian Zou та Le Dong. «Cryptanalysis of the round-reduced Kupyna hash function». B: *Cryptology ePrint Archive* (2015).

ДОДАТОК А ВЕЛИКІ РИСУНКИ ТА ТАБЛИЦІ

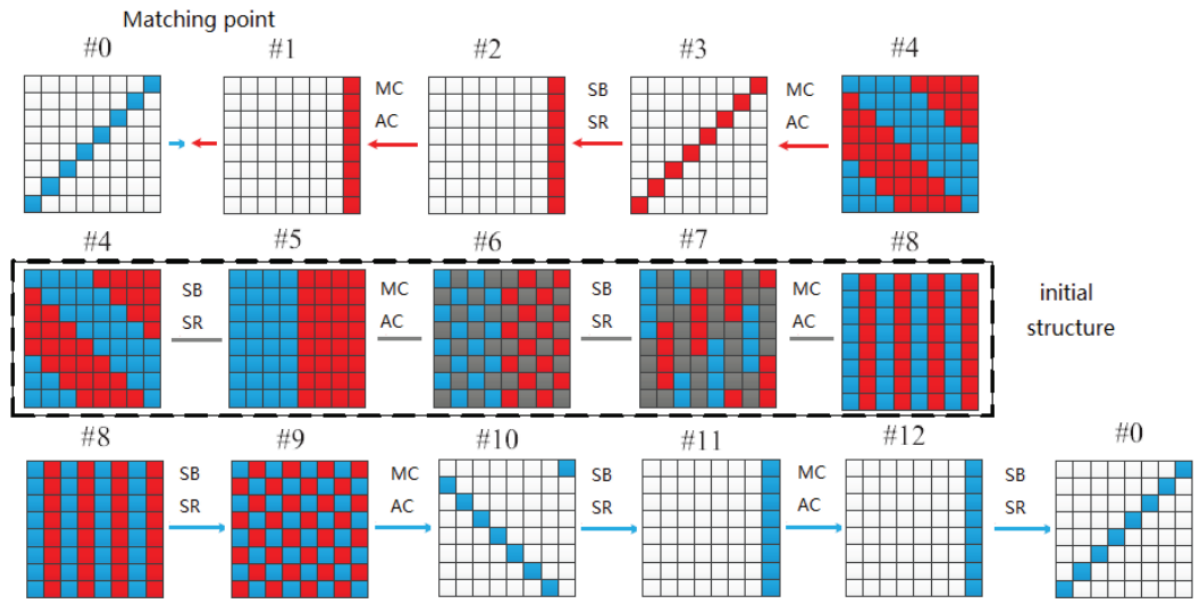


Рисунок А.1 – Розділення фрагментів підпростору атаки прообразу на 6-раундовий алгоритм Кируна-256 [17]

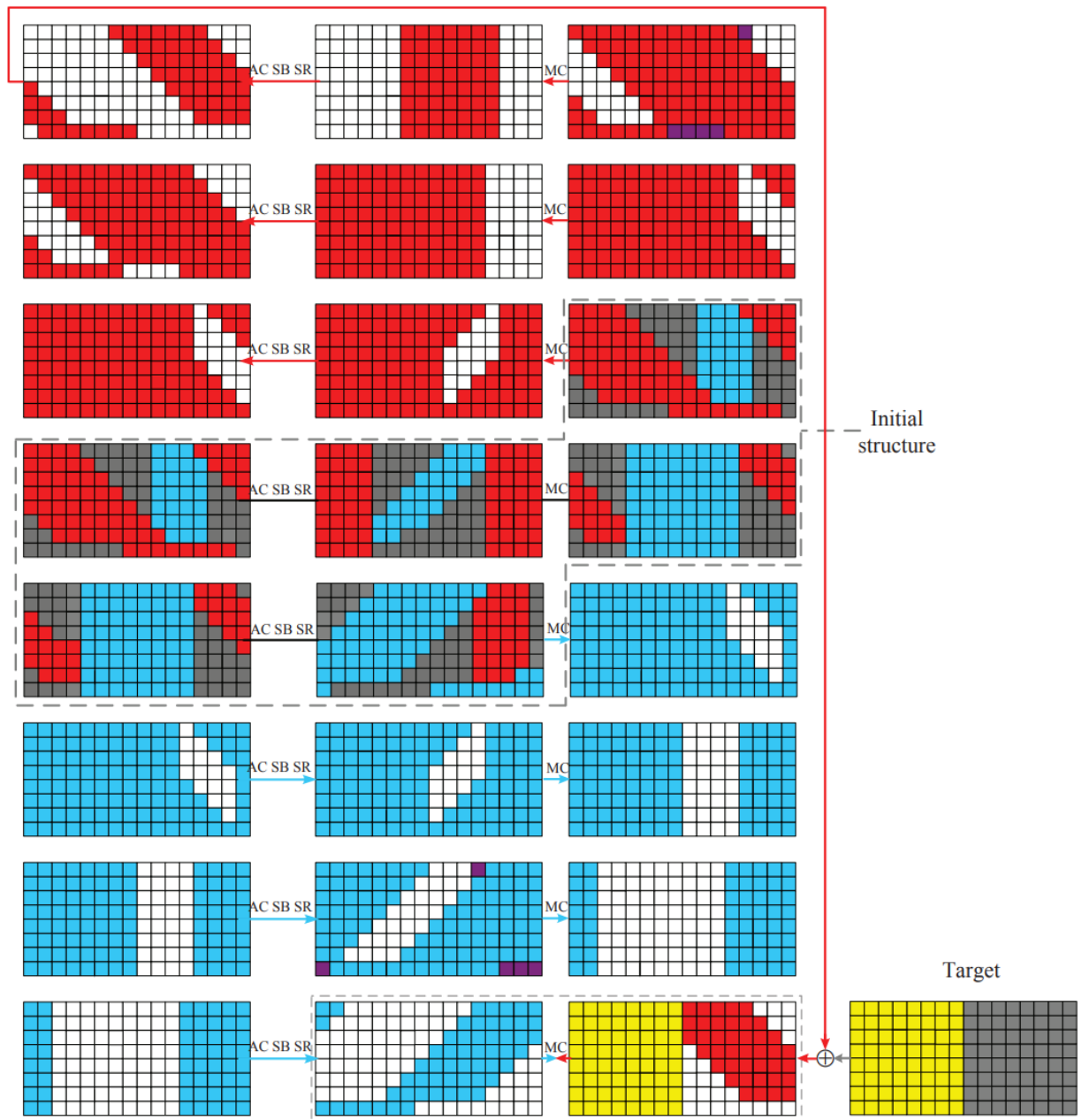


Рисунок А.2 – Розділення атаки прообразу на 8-раундовому Курупа-512

[17]

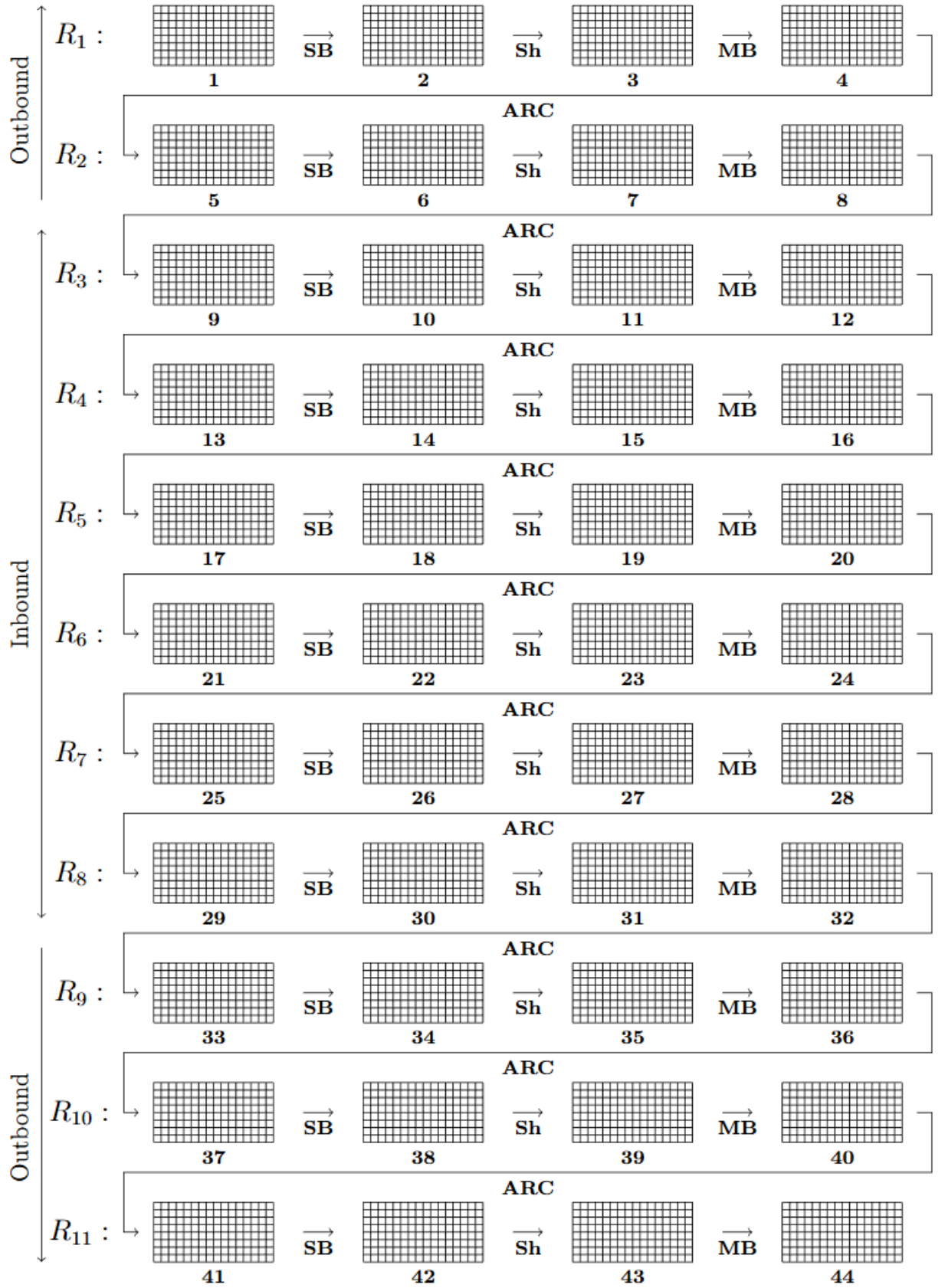


Рисунок А.3 – 11-раундовий усічений диференційний шлях на
Grøstl-512 [1]

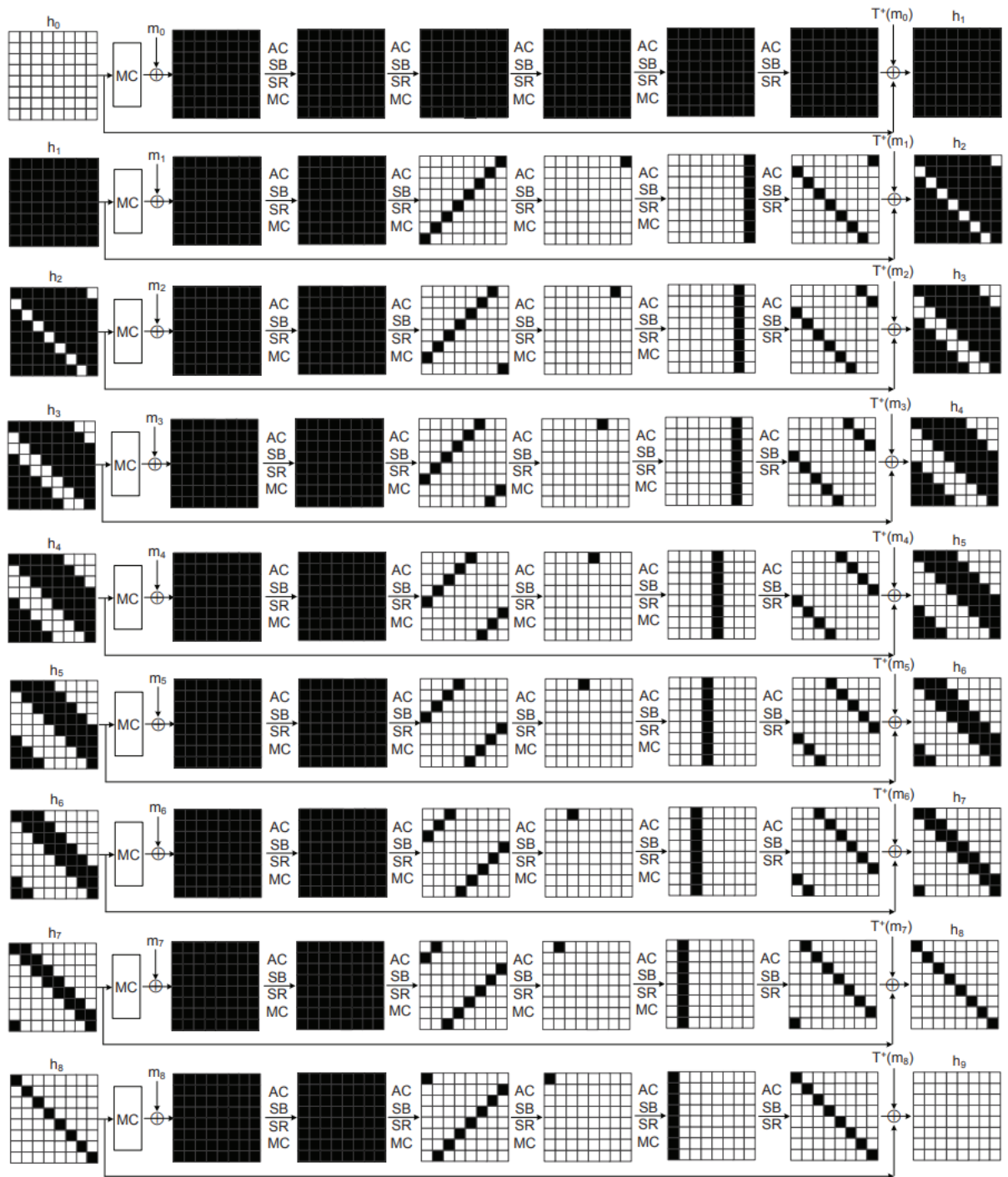


Рисунок А.4 – Укорочений диференціальний шлях, використаний під час атаки на 5 раундів «Купини» [17]

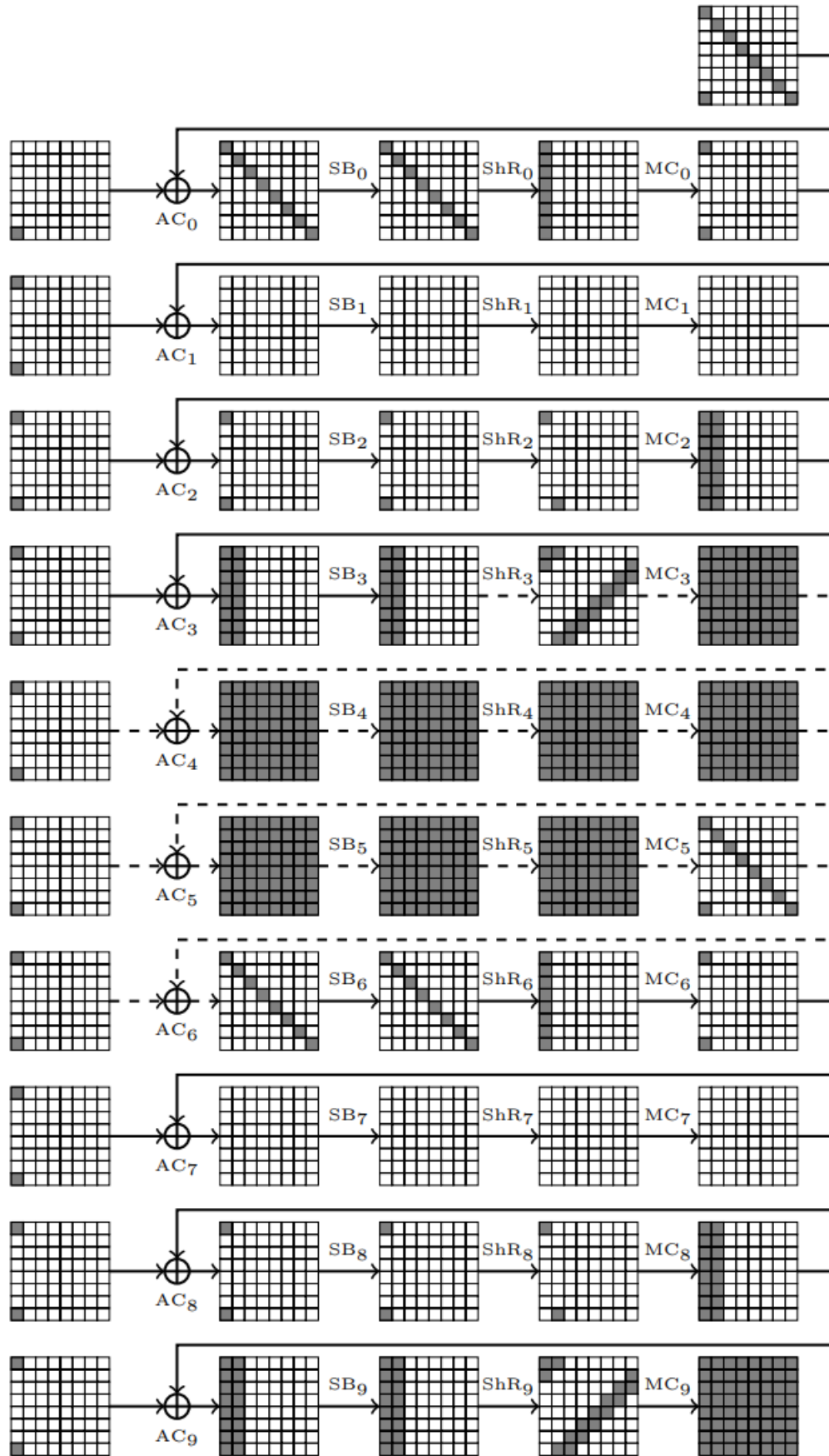


Рисунок А.5 – 10-раундовий диференціальний шлях між P і Q для алгоритму Grøstl-256. Кожна комірка відповідає одному байту, а сіра комірка позначає активний байт [14]