Кібернетика та комп'ютерні технології

УДК 519.6 DOI:10.34229/2707-451X.26.1.8

І.А. БАРАНОВ

КРИПТОГРАФІЧНИЙ АНАЛІЗ СКЛАДНОСТІ ТА ТЕСТУВАННЯ СИМЕТРИЧНОГО БЛОЧНОГО АЛГОРИТМУ WBC2

Описується симетричний блочний криптографічний алгоритм WBC2. Розглядається процес шифрування, аналіз складності алгоритму і швидкості виконання. Показана реалізація та результати тестування алгоритму.

Ключові слова: *симетричний блочний криптографічний алгоритм, аналіз складності симетричного блочного криптоалгоритму, аналіз швидкості криптоалгоритму, S-box, дифузія, раундовий ключ, NIST Statistical Test Suite.*

Вступ. Комп'ютерні мережі на сьогодні набувають дедалі більшого значення для обміну інформацією, а криптографія відіграє ключову роль у гарантуванні безпеки обчислень, захисті зв'язку, паролів та критично важливої інформації в цифрових системах. Основна мета криптографії – зробити конфіденційну інформацію недоступною для сторонніх осіб, залишаючи її читабельною лише для передбачуваного отримувача.

Вагомий внесок у розвиток української криптографічної науки здійснили В.К. Задірака та А.М. Кудін [1–7], а також, І.Д. Горбенко та А. Кузнецов [8–10], наукові праці яких присвячені дослідженню шифрів та їх безпеки. Сучасні національні стандарти України в галузі криптографії [11–16] забезпечують високий рівень стійкості до атак та сумісність із квантово-захищеними підходами, що відповідає сучасним викликам кібербезпеки.

Найбільшого розвитку та застосування в системах захисту інформації набули симетричні криптографічні перетворення [17–19]. Наприклад, український національний стандарт ДСТУ 7624:2014 «Калина» [11, 20], розроблений під керівництвом професорів І. Горбенка та А. Кузнецова – це сучасний симетричний блоковий шифр, що забезпечує високий рівень безпеки та стійкості до атак. У 2015 році його було прийнято як державний стандарт шифрування України. Нині алгоритм широко застосовується в різних сферах, включаючи дослідження інформаційної безпеки та криптографії. Розробка ДСТУ 7624:2014 базується на відомому шифрі AES [21], але вносить деякі суттєві модифікації, що покращують його властивості безпеки. Однією з основних відмінностей між ДСТУ 7624:2014 та AES є використання різних випадково згенерованих S-блоків, замість одного фіксованого S-блоку, що використовується у кожному раунді AES.

Рівень захищеності інформаційних ресурсів залежить не лише від властивостей застосовуваного блокового симетричного шифру (БСШ), але й від способів його використання, тобто властивості криптографічного перетворення безпосередньо залежать від режиму застосування [22, 23].

Статистичний тестовий набір NIST – це важливий тестовий набір для аналізу випадковості, який часто використовується для офіційних сертифікацій або затверджень. Слід зауважити, що тестування алгоритмів за методиками NIST STS [24] охоплює оцінку стійкості до атак, ефективності обчислень, якості генерації ключів та здатності забезпечувати конфіденційність даних у різних сценаріях використання. Результати таких тестів підтверджують надійність алгоритмів, їх відповідність міжнародним вимогам безпеки та дозволяють визначити оптимальні параметри для практичного застосування у промислових і державних інформаційних системах.

Сучасні розробки у сфері блочно-орієнтованих алгоритмів спрямовані на підвищення продуктивності, зниження обчислювальних витрат та інтеграцію у квантово-захищені національні стандарти. Для посилення стійкості алгоритму WBC1 [25] у роботі пропонується його модифікація – алгоритм WBC2 (White-Box Cube Cipher 2). WBC2 є сучасним симетричним блочним шифром, що розширює модель WBC1 до більш стійкої, нелінійної та гнучкої структури. Алгоритм базується на моделі куба Рубіка, що дозволяє створювати унікальні операції на основі ключа.

1. Алгоритм WBC2. У криптографії термін «білий ящик» (white-box) описує модель, у якій алгоритм функціонує в умовах повної прозорості: зловмисник має доступ до коду, пам'яті, проміжних значень, таблиць та структур даних. На відміну від моделі «чорного ящика», де атакуючий бачить лише вхід і вихід алгоритму, white-box-сценарій потребує додаткових механізмів захисту, оскільки вся реалізація є відомою.

WBC2 розроблено з урахуванням зазначених загроз і реалізує індивідуалізацію всіх внутрішніх перетворень (зокрема, S-box та перестановок) від ключа. Це означає, що навіть якщо зловмисник володіє повним кодом алгоритму, всі таблиці, оператори і структура операцій будуть унікальними для кожного ключа. Таким чином, у WBC2 відсутні фіксовані загальні таблиці або уніфіковані операції – шифрування повністю визначається секретним ключем.

Попри те, що WBC2 не є повноцінним white-box-шифром у класичному розумінні (де ключ має залишатися прихованим навіть при повному доступі до реалізації), він використовує white-box-принципи індивідуалізації, що значно ускладнює аналіз навіть у відкритому середовищі.

1.1. Процес шифрування. Нехай вхідний відкритий текст T розбивається на блоки B_i довжиною $s = d^3$ байтів, де d – розмір сторони куба. Позначимо блок у вигляді $B = (b_0, b_1, .. b_{s-1})$. Ключ $K = \{0,1\}^{256}$ (256 біт, 32 байти).

Для кожного ключа K генерується таблиця перестановок (таблиця операцій)

$$P = (p_0, p_1, .. p_{126}),$$

де p_i – унікальна операція над кубом: обертання грані, зрізу, шаблон або динамічна послідовність. Таблиця P формується на основі ключа та залишається незмінною для заданого K. На кожному раунді з таблиці обирається операція за індексом, що обчислюється як функція від ключа та номера раунду:

$$index_r = f_K(r),$$
$$\pi_r = P(index_r),$$

де π_r – перестановочна операція r-го раунду.

Також в алгоритмі WBC2 було додано S-box, яка є динамічною та залежить від секретного ключа. Вона перетворює кожен байт (значення від 0 до 255) за деякою перестановкою. S-box підвищує плутанину (confusion) між входом та виходом, ускладнюючи аналіз шифру. Для її побудови використовується такий підхід:

На першому кроці будується базова лінійна таблиця

$$S[i] = i, \quad i = 0, \dots, 255.$$

Далі з ключа K генерується сид для генератора

```
rng = random.Random()
rng.seed(hashlib.sha256(key).digest())
```

Для перемішування таблиці застосовується метод Фішера – Йетса для псевдовипадкової перестановки

```
for i in range(255, 0, -1):
    j = rng.randint(0, i)
    sbox[i], sbox[j] = sbox[j], sbox[i]
```

Таким чином формується ключезалежна бієкція S: $\{0, \dots, 255\} \to \{0, \dots, 255\}$.

Для розшифрування формується зворотна S-box. Для цього обчислюється зворотна таблиця S^{-1}):

```
inverse_sbox = [0]*256
for i, val in enumerate(sbox):
    inverse_sbox[val] = i
```

Отриманий S-box є унікальним для кожного ключа та зберігає властивість бієкції. Це забезпечує як криптографічну стійкість, так і відновлюваність.

Розглянемо опис одного раунду шифрування. Нехай вхідним даними r-го раунду є $X^{(r)}$ (на першому раунді $X^{(1)} = B$), r – номер поточного раунду, де $r = 1,2, \dots, R$, R – загальна кількість раундів в алгоритмі.

1. Перестановка куба

$$\mathrm{X}_1 = \pi_r(X^{(r)}),$$

де π_r – операція з індивідуалізованої таблиці перестановок P.

2. XOR з раундовим ключем. До кожного байта (або елемента) результату перестановки X_1 застосовується операція XOR з відповідним байтом раундового ключа RK_r:

$$\mathrm{X}_2 = \mathrm{X}_1 \oplus RK_r,$$

де RK_r – раундовий ключ (псевдовипадкова послідовність байтів, що залежить від K та r). Цей етап забезпечує зв'язок між даними, що шифруються, і секретним ключем, підвищуючи стійкість.

3. S-box. Кожен байт (або елемент) $\mathrm{X}_2[i]$ пропускається через S-box – нелінійну таблицю замін, індивідуалізовану за ключом:

$$\mathrm{X}_3[i] = S(\mathrm{X}_2[i]), i = 0,1, \dots, s-1,$$

де S – індивідуалізована S-box (бієкція $\{0, \dots, 255\}$), визначена ключем K. S-box генерується для кожного ключа окремо і може бути різним для різних запусків. Цей крок забезпечує виражену нелінійність та захист від лінійного та диференціального криптоаналізу.

4. Дифузія (накопичувальний XOR). Перший елемент Y_0 дорівнює $\mathrm{X}_3[0]$. Кожен наступний елемент Y_i обчислюється як XOR поточного значення $\mathrm{X}_3[i]$ та попереднього результату Y_{i-1}:

$$Y_0 = \mathrm{X}_3[0],$$
$$Y_i = \mathrm{X}_3[i] \oplus Y_{i-1}, i = 1, \dots, s-1,$$
$$\mathrm{X}_4 = (Y_0, Y_1, \dots, Y_{s-1}).$$

Така ланцюгова операція поширює вплив кожного байта на всі наступні байти, збільшуючи дифузію (ефект лавини).

5. Циклічний побітовий зсув

$$\mathrm{X}_5[i] = ROTR_{n_r}(\mathrm{X}_4[i]),$$

де параметр n_r – кількість бітів зсуву, що визначається ключем та номером раунду.

6. Перехід до наступного раунду

$$X^{(r+1)} = \mathrm{X}_5.$$

Після виконання R раундів результат шифрування набуває вигляду:

$$C = X^{(R+1)}.$$

Розглянемо опис одного раунду розшифрування (зворотний раунд)

1. Зворотний циклічний побітовий зсув

$$\mathrm{X}_4[i] = ROTL_{n_r}(\mathrm{X}_5[i]).$$

2. Зворотна дифузія

$$Y_{s-1} = \mathrm{X}_4[s-1],$$
$$Y_{i-1} = \mathrm{X}_4[i-1] \oplus Y_i,\ i = s-1, \dots, 1,$$
$$\mathrm{X}_3 = (Y_0, Y_1, \dots, Y_{s-1}).$$

3. Зворотна S-box

$$\mathrm{X}_2[i] = S^{-1}(\mathrm{X}_3[i]).$$

4. XOR з раундовим ключем

$$X_1 = \mathrm{X}_2 \oplus RK_r.$$

5. Зворотна перестановка

$$X^{(r)} = \pi_r^{-1}(\mathrm{X}_1).$$

Нехай F_R – функція r-го раунду, яка складається з послідовної композиції всіх описаних перетворень (перестановка, XOR з ключем, S-box, дифузія, бітовий зсув). Тоді повна схема шифрування описується як

$$C = F_R \circ F_{R-1} \circ \cdots \circ F_1(B),$$

де B – відкритий текст, C – зашифрований текст (ciphertext), R – кількість раундів.

Для зворотного перетворення використовуються обернені раундові функції

$$B = F_1^{-1} \circ F_2^{-1} \circ \cdots \circ F_R^{-1}(C).$$

Порядок обернених функцій зворотний до шифрування: перша застосовується F_R^{-1}, потім F_{R-1}^{-1} і т.д.

Всі внутрішні структури алгоритму є ключозалежними та унікальними для кожного ключа. Навіть при повному доступу до коду та таблиць, атакуючий не може відновити ключ без повного перебору простору ключів.

На відміну від теоретичної моделі з довільними перестановками, алгоритм WBC2 використовує динамічну ключозалежну таблицю перестановок, що забезпечує лінійну обчислювальну складність відносно розміру даних та кількості раундів у практичній реалізації. Динамічна генерація таблиці перестановок та індивідуалізація S-box для кожного ключа значно підвищує криптографічну стійкість алгоритму. Навіть за обмеженого базового набору перестановочних операцій їх ключозалежна комбінація в кожному раунді формує експоненційно велику кількість різних сценаріїв шифрування. У моделі повного розкриття (white-box), де зловмисник має доступ до всіх структур алгоритму, простір перебору залишається надзвичайно великим завдяки індивідуалізації всіх таблиць і операцій для кожного ключа. Запропонована модель дозволяє змінювати кількість операцій у таблиці перестановок, параметри генерації S-box, кількість раундів, що забезпечує можливість тонкого налаштування рівня безпеки та продуктивності під конкретні задачі.

1.2. Аналіз складності алгоритму. Нехай n – загальний розмір вхідних даних у бітах; s – розмір одного блоку в бітах $s = d^3$ байт, де d – розмір сторони куба; k – кількість блоків, $k = \frac{n}{s}$; R – кількість раундів шифрування; m – довжина секретного ключа у бітах (зазвичай 256); N_{op} – кількість унікальних операцій у таблиці перестановок, Q_P – кількість можливих варіантів побудови таблиці перестановок; Q_{RK} – простір можливих наборів раундових ключів і параметрів зсуву; S – ключозалежна S-box (бієкція на 256 елементів).

Опишемо складність алгоритму шифрування. Процес шифрування включає такі етапи:

1. Розподіл вхідних даних на блоки – це $O\left(\frac{n}{s}\right)$ операції.

2. Для кожного блоку виконується застосування перестановок – $O(s)$; XOR з раундовим ключем – $O(s)$; S-box – $O(s)$; дифузія – $O(s)$; циклічний побітовий зсув – $O(s)$.

Оскільки всі операції для одного блоку мають лінійну складність $O(s)$, то для всіх R раундів сумарна складність для одного блоку буде $O(R \cdot s)$, для всіх блоків $O(k \cdot R \cdot s) = O(n \cdot R)$.

Одноразова ініціалізація ключових структур в алгоритмі WBC2 виконується перед початком обробки даних і включає кілька ключових етапів, що залежать від секретного ключа K. Першим кроком є генерація таблиці перестановок P, яка складається з N_{op} унікальних перестановок (наприклад, 127). Кожен елемент p_i у цій таблиці є елементарною або складеною операцією над блоком розміру s, і вся таблиця створюється детерміновано з використанням ключа, наприклад, через псевдовипадковий генератор із сідом від K. Обчислювальна складність цього етапу становить $O(N_{op} \cdot s)$, оскільки для кожної перестановки потрібно підготувати масив індексів.

Другий етап – побудова ключезалежної S-box – бієкції на множині з 256 елементів $S: \{0, \dots, 255\} \to \{0, \dots, 255\}$, яка генерується за допомогою алгоритму Фішера – Йетса з використанням хешу ключа як початкового сідового значення. Складність цього етапу – $O(256)$. Після цього формується обернена таблиця S^{-1}, необхідна для розшифрування – також за $O(256)$.

Наступний етап – це генерація раундових ключів RK_r і параметрів циклічного зсуву n_r для кожного з R раундів. Кожен RK_r може бути отриманий, наприклад, шляхом хешування комбінації ключа і номера раунду або за допомогою функції розтягування ключа (KDF). Параметр зсуву nrn_rnr також обчислюється як функція від K і r. На кожен раунд припадає складність $O(s)$ для генерації масиву байтів ключа і $O(1)$ для зсуву, що загалом дає $O(R \cdot s)$.

Підсумкова обчислювальна складність усієї ініціалізації має вигляд

$$T_{init} = O(N_{op} \cdot s + 256 + R \cdot s).$$

Ці витрати виконуються лише один раз при старті шифрування для кожного нового секретного ключа. Для типових параметрів $N_{op} = 127$, $s = 64$, $R = 10$ витрати на ініціалізацію є незначними порівняно з витратами на обробку великих обсягів інформації. Ініціалізація забезпечує унікальність усіх внутрішніх структур та криптографічну стійкість алгоритму WBC2 для кожного ключа, що є критично важливим для захисту в моделі white-box.

Загальна обчислювальна складність алгоритму WBC2

$$O(n \cdot R) + O(N_{op} \cdot s + 256 + R \cdot s).$$

Компоненти простору перебору в алгоритмі WBC2 описуються формулою

$$S_{script} = 2^m \cdot N_{op}^R \cdot Q_P \cdot 256! \cdot Q_{RK}.$$

Ця формула відображає загальну кількість унікальних конфігурацій шифру, які повинен перебрати атакувальник для гарантованого відновлення всіх внутрішніх параметрів алгоритму. Розглянемо детально кожен множник.

2^m – простір секретних ключів. Якщо довжина ключа становить m біт (наприклад, 256), то існує 2^m можливих варіантів ключа K.

N_{op}^R описує кількість можливих послідовностей перестановок у раундах. Тут N_{op} – кількість унікальних операцій у таблиці перестановок P, наприклад, 127. У кожному з R раундів алгоритм вибирає одну перестановку з таблиці. Навіть якщо таблиця фіксована, конкретна послідовність її використання залежить від ключа. Отже, кількість можливих комбінацій перестановок становить N_{op}^R.

Q_P – простір можливих таблиць перестановок. Таблиця P може генеруватися на основі ключа з набору базових операцій, шаблонів або динамічних правил. Якщо таблиця однакова для всіх ключів,

тоді $Q_P = 1$. Якщо ж таблиця формується з великого набору підмножин кількість варіантів Q_P може бути надзвичайно великою.

256! – це кількість усіх можливих S-box. Для кожного ключа створюється унікальна бієкція на множині з 256 елементів. Навіть якщо атакувальник знає алгоритм генерації, визначити ключ на основі відомої S-box майже неможливо через колосальну кількість варіантів – 256!.

Множник Q_{RK} описує простір усіх можливих комбінацій раундових ключів RK_r і параметрів зсуву n_r. Навіть якщо ці параметри повністю визначаються основним ключем *K*, способів їх побудови для кожного ключа може бути значною, особливо якщо застосовується стохастичне або параметризоване генерування.

Для успішної атаки на алгоритм недостатньо знати лише один із параметрів. Необхідно відновити повний набір: секретний ключ, конкретну послідовність перестановок, конкретну таблицю перестановок, конкретну S-box, а також усі раундові ключі та параметри зсувів. Оскільки всі ці структури незалежні й генеруються індивідуально для кожного ключа, їхній спільний простір перебору утворюється як добуток відповідних потужностей множин.

Таким чином, форується астрономічно великий простір можливих реалізацій алгоритму WBC2, що забезпечує його криптографічну стійкість навіть у моделі з повним розкриттям (white-box).

1.3. Швидкість виконання алгоритму. Загальний час виконання алгоритму можна подати у вигляді:

$$T_{total} = T_{split} + T_{init} + T_{encrypt} + T_{decrypt},$$

де T_{split} – час для розділення даних, T_{init} – час одноразової ініціалізації ключових структур, $T_{encrypt}$ – час шифрування даних, $T_{decrypt}$ – час розшифрування даних.

Розглянемо кожен з доданків:

$$T_{split} = O(n),\ T_{init} = O(N_{op} \cdot s + 256 + R \cdot s),$$
$$T_{encrypt} = T_{decrypt} = O(n \cdot R).$$

Отже, загальний час виконання алгоритму має вигляд

$$T_{total} = O(n) + O\big(N_{op} \cdot s + 256 + R \cdot s\big) + 2 \cdot O(n \cdot R).$$

Розглянемо приклад. Розмір вхідних даних: n = 1 000 000 біт, розмір блоку s = 64 біт, кількість блоків $k = \frac{n}{s} = \frac{1{,}000{,}000}{64} \approx 15{,}625$, кількість раундів $N_{op} = 127$, кількість операцій у залежності від ключа m = 256.

Підставимо дані у формулу

$$T_{total} = O\big(1\,000\,000\big) + O(127 \cdot 64 + 256 + 64 \cdot 64) + 2 \cdot O\big(1\,000\,000 \cdot 64\big),$$

$$T_{total} \approx O(1{,}29 \cdot 10^8).$$

Алгоритм WBC2 має лінійну складність за обсягом даних та кількістю раундів, дозволяє ефективно обробляти великі масиви інформації:

$$T_{WBC2}(n, R) = O(n \cdot R).$$

Основні операції кожного раунду (перестановка, XOR, S-box, дифузія, циклічний зсув) легко реалізуються та оптимізуються на сучасних процесорах. Час одноразової ініціалізації ключових структур (таблиця перестановок, S-box, раундові ключі) є незначним порівняно з основним процесом шифрування, особливо під час роботи з великими обсягами даних.

На відміну від WBC1, де використання довільної перестановки призводить до експоненційного зростання складності, WBC2 забезпечує оптимальне співвідношення між захищеністю та продуктивністю.

1.4. Стійкість до методів криптоаналітичних атак. Алгоритми WBC2, як і WBC1, стійкий до кількох видів криптоаналітичних атак завдяки своїй унікальній структурі та використовуваним методам. Розглянемо основні методи стійкості.

Стійкість до перебору (Brute Force). Перебір – один із базових і найбільш відомих методів криптоаналізу [26 – 29], за якого криптоаналітик послідовно перевіряє всі можливі ключі, доки не буде знайдено правильний. Цей метод особливо ефективний проти алгоритмів із короткими ключами, оскільки збільшення довжини ключа експоненційно підвищує кількість можливих комбінацій. На відміну від WBC1, де експоненційна стійкість забезпечується за рахунок великого простору перестановок, у WBC2 експоненційна стійкість закладена в поєднанні значної довжини ключа (наприклад, 128 або 256 біт), динамічної генерованої ключозалежної таблиці перестановок, ключозалежної S-box (бієкції), унікальних раундових ключів і параметрів зсуву. Простір перебору для атакувальника у WBC2 визначається як добуток усіх варіантів, де кожен множник відповідає кількості можливих варіантів ключа, перестановок, S-box тощо. Це робить атаку повного перебору практично неможливою навіть за наявності повного доступу до коду (white-box модель).

Диференціальний криптоаналіз. Диференціальний криптоаналіз був розроблений Елі Біхамом і Аді Шаміром [29, 30] і є одним із найвідоміших методів криптоаналізу, особливо для блочних шифрів. Цей метод дає змогу виявляти залежності між різницями вхідних і вихідних даних, що дозволяє розкрити структуру шифру та в окремих випадках підібрати ключ. Алгоритм WBC2 використовує багатократний раундовий процес, у якому для кожного раунду застосовуються унікальні ключозалежні перестановки, S-box, дифузійні операції (накопичувальний XOR) та циклічні побітові зсуви. Таке поєднання забезпечує високу ентропію на кожному раунді та складну нелінійну взаємодію між бітами вхідних і вихідних даних, що ускладнює прогнозування впливу різниць у відкритому тексті на шифротекст. Крім того, ключозалежна S-box генерується для кожної сесії окремо, що унеможливлює попередню підготовку атак або використання готових диференціальних структур.

Лінійний криптоаналіз [31–33] – метод криптоаналізу, запропонований Міцуру Мацуї [31] у 1993 році, призначений для атаки на блокові шифри. Основна ідея цього методу полягає у пошуку лінійних апроксимацій між відкритим текстом, шифротекстом та ключем, що дозволяє криптоаналітику визначати ключ на основі відповідних ймовірнісних залежностей. Завдяки використанню складних ключозалежних перестановок в алгоритмі WBC2, нелінійної S-box, дифузії та циклічних зсувів, алгоритм формує значний рівень випадковості та нелінійності. Кожен раунд унікальний для конкретного ключа, що унеможливлює застосування універсальних лінійних апроксимацій. Динамічна генерація S-box додатково ускладнює пошук статистичних закономірностей. Отже, алгоритм WBC2 має високу стійкість до лінійного криптоаналізу, оскільки операції ускладнюють побудову ефективних лінійних залежностей.

Атаки на базі сайд-каналів [34–36] – це метод криптоаналізу, за якого зловмисник використовує фізичну інформацію, отриману під час роботи криптографічного пристрою, для вилучення секретної інформації, зокрема ключів шифрування. Такі атаки грунтуються не на прямому аналізі алгоритму шифрування, а на дослідженні додаткових даних, що виникають у процесі виконання. Використання складної структури раундових операцій в алгоритмі WBC2, значної кількості раундових ключів, а також ключозалежних S-box і перестановок ускладнює кореляцію між обробкою даних і зміною апаратних характеристик. Багатоетапна нелінійність і динамічність внутрішніх параметрів можуть знижувати ефективність класичних сайд-канальних атак. Однак повна захищеність від таких атак потребує додаткових досліджень і апаратного захисту.

На відміну від алгоритмів на кшталт «Калина», стійкість яких ґрунтується на доведених властивостях нелінійних S-блоків у скінченних полях, механізм захисту WBC2 базується на експоненційно

великому просторі ключозалежних операцій перестановки та унікальних для кожного ключа S-блоках. Це робить класичні методи диференціального та лінійного криптоаналізу, ефективні проти стандартних SP-мереж, практично непридатними.

Загалом, як відомо, досконалої криптографічної стійкості не існує. Поєднання в алгоритмі WBC2 численних раундових ключозалежних структур, дифузійних операцій і нелінійних перетворень забезпечує високий рівень криптографічної стійкості навіть у жорстких умовах white-box моделі, де атакувальник має повний доступ до програмної реалізації шифру.

1.5. Апробація алгоритму. Алгоритм WBC2 відповідно до ISO/IEC 10116-2006 [22], ISO/IEC 10116:2017 [23] було реалізовано для таких режимів роботи: ECB (Electronic Codebook), CBC (Cipher Block Chaining), CFB (Cipher Feedback), OFB (Output Feedback), CTR (Counter mode), WBC-CTR-HMAC (потоковый режим с HMAC). У лістингу 1 наведено фрагмент програми Rubikrypt [37], який описує один раунд шифрування за алгоритмом WBC2 у режимі ECB. У лістингу 2 показано протокол виконання програми шифрування та розшифрування тексту в режимі ECB.

Лістинг 1. Фрагмент програми Rubikrypt, який описує один раунд шифрування за алгоритмом WBC2 у режимі ECB

```
def _encrypt_block(self, block: bytes, block_size: int):
        cube = self._form_cube(block, block_size)
        cube_bytes = cube.size
        rounds = self.rounds or 32
        for round_number in range(rounds):
            round_key = self._get_round_key(round_number, cube_bytes)
            op_id = round_key[0] % len(self.operations)
            cube = self._apply_operation(cube, op_id, inverse=False)
            xor_bytes = np.frombuffer(round_key, np.uint8)
            flat = cube.flatten()
            flat = (flat ^ xor_bytes).astype(np.uint8)
            flat = self.sbox[flat]
            # Диффузия: накопительный XOR (forward)
            for i in range(1, len(flat)):
                flat[i] ^= flat[i-1]
            cube = flat.reshape(cube.shape)
            cube = _bitwise_rotate_cube(cube, op_id, 'right')
        return cube.tobytes(), block_size
```

Лістинг 2. Протокол виконання програми WBC2 у режимі ECB (шифрування та розшифрування)

```
=== RUBIKRYPT CIPHER WBC2 ===
1. Encrypt/decrypt text
2. Show rotation operations table
3. Run self-tests
4. Benchmark performance
5. Generate NIST test data
6. Differential and avalanche tests + statistics/graphs
7. Exit
Select mode (1-7): 1
```

```
Enter text to encrypt: Унікальність і новаторство в алгоритмі WBC1, що відрізняє
його від інших відомих алгоритмів, полягає в тому, що в даному криптоалгоритмі
дані представлені в тривимірному просторі

Select the cipher operating mode:
1. ECB (Electronic Codebook)
2. CBC (Cipher Block Chaining)
3. CFB (Cipher Feedback)
4. OFB (Output Feedback)
5. CTR (Counter Mode)
6. WBC-CTR-HMAC (author special mode)
Mode [1-6, по умолчанию 1]: 1

Generate key automatically? (y/n): y

Generated                                                                  key:
7008453c139571650f15d49747233da781d19e767ca69d829182bb0a959dad15 (save this for
decryption)

Enter the number of rounds (default 32): 32

Encrypted data (HEX):
1338b1e165b48356d4eb08d151a432d3084a24715139f4ea91a479fefe835f9b0099e342c0da48
f4935106c6ae5bcfef4c41aadc5639bd9443ba69e0012ba1323bcbfe31c2319059c73ffc5328f9
5f84f25facf15b107915ba4a3e2a45aa9a492cec93f7bdc04b403dd2d8170ca82094a069cc0402
488ba71575aa8bb05d1056cc9de585e08aecb7b53fb03ce96b13613acf5979af41f6cd977d4d87
20f7645a68e5249b46e76af6138e56fb1ad7d1c9349ef705d7603e40a4120cde529e3630c7a942
daaf68d5dadfc23174b786d6d5bf1bf37999151fab2d961da95ee3598d800c3a4d86c2bca719ce
173a1f2a9e4655f237c7442ee35c4b2a9384da444790c344c4c398ad5e01f8e0f424c1ad6080f4
b97a666652636758e339bca24d6742d49bed08b30ee6a12e7f22329aa398c030b1b79681c5cea1
e30e9a445b82d1cedebf8466328faa26b3a6d245fd12077409b17fe2db8186

Decrypted text: Унікальність і новаторство в алгоритмі WBC1, що відрізняє його
від інших відомих алгоритмів, полягає в тому, що в даному криптоалгоритмі дані
представлені в тривимірному просторі
```

На рисунку показано залежність часу виконання алгоритмів WBC1 і WBC2 у різних режимах роботи й за різних обсягів даних (32 раунди). Порівнюючи час виконання алгоритмів WBC1 і WBC2, можна зазначити, що WBC1 працює помітно швидше. Основна причина полягає в особливостях конструкції алгоритмів. В WBC2 було додано додаткові операції, спрямовані на посилення лавинного ефекту (тобто суттєвої зміни шифротексту за мінімальної зміни відкритого тексту або ключа). З цією метою в WBC2 реалізовано додаткові раундові перетворення, розширені S-box і дифузійні процедури. Такі ускладнення підвищують криптостійкість, однак водночас збільшують обчислювальне навантаження на кожному раунді. WBC1, навпаки, використовує більш лаконічний набір перетворень, які виконуються швидше завдяки оптимізації коду та меншій кількості внутрішніх операцій. Саме тому WBC1 демонструє вищу продуктивність під час шифрування великих обсягів даних. Отже, збільшенням часу обробки у WBC2 є платою за підвищену криптографічну складність і посилений лавинний ефект. Це типовий компроміс між швидкодією та рівнем захисту: чим більше захисних механізмів, тим більше обчислень і повільніше загальна робота алгоритму.

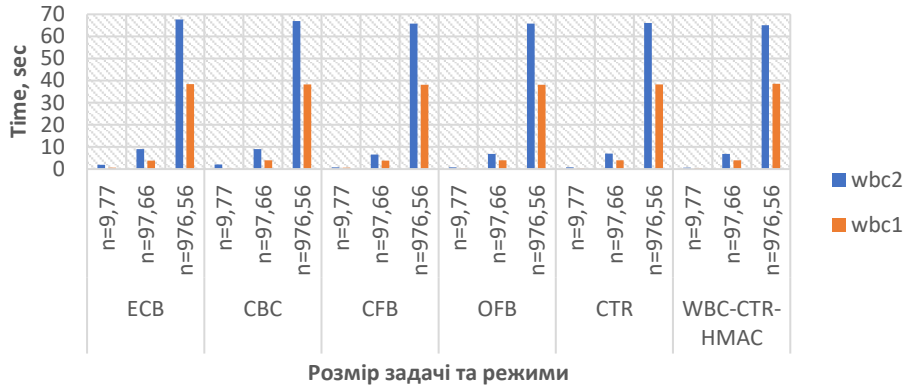

РИСУНОК. Порівняння часу виконання програми WBC1 та WBC2 залежно від розміру задачі, та режиму роботи (32 раунди)

Було проведено комплексне тестування алгоритму WBC2 з використанням автоматично згенерованого 256-бітного ключа та 64 раундів шифрування. Оцінювання включало диференціальний тест, лавиннтй ефекту, а також статистичну перевірку ентропії, розподілу байтів і чутливості до ключа. Результати тестування наведено в лістингу 3.

Лістинг 3. Комплексне тестування алгоритму WBC2 з використанням автоматично згенерованого 256-бітного ключа та 64 раундів шифрування

```
  === RUBIKRYPT CIPHER WBC2 ===
  1. Encrypt/decrypt text
  2. Show rotation operations table
  3. Run self-tests
  4. Benchmark performance
  5. Generate NIST test data
  6. Differential and avalanche tests + statistics/graphs
  7. Exit
  Select mode (1-7): 6
  === Differential and avalanche tests + statistics/graphs ===
  Enter test text (random data by default):
  Select the cipher mode:
  1. ECB (Electronic Codebook)
  2. CBC (Cipher Block Chaining)
  3. CFB (Cipher Feedback)
  4. OFB (Output Feedback)
  5. CTR (Counter Mode)
  6. WBC-CTR-HMAC (author special mode)
  Mode [1-6, по умолчанию 1]: 2
  Generate key automatically? (y/n): y

  Generated key:
a266c04b84ca4cc7a3f79659995e31dedda62391ba520d0c47944b4e4511dace
(save this for decryption)

  Enter the number of rounds (default 32): 64
```

```
Avalanche effect: 50.09% of output bits change on single input bit flip.
Differential test: 50.05% of output bits change on key bit flip.

=== STATISTICAL TESTS ===
Shannon entropy (plain):  5.6875 bits/byte
Shannon entropy (cipher): 6.4177 bits/byte
Chi-square (ciphertext):  300.00
Correlation (plain↔cipher): nan
Repetition in cipher:     1 adjacent bytes repeated
Repeated 8-byte blocks:   0
Key sensitivity:          50.11% of output bits change on key bit flip.
```

Результати експериментального тестування алгоритму WBC2 у режимі CBC (Cipher Block Chaining) підтверджують його відповідність сучасним криптографічним вимогам. Однією з ключових ознак надійного шифру є лавинний ефект, у реалізації WBC2 він досягає оптимального рівня – близько 50 % змінених бітів за модифікації одного біта вхідних даних, що свідчить про високий ступінь дифузії всередині алгоритму. Крім того, алгоритм демонструє високу чутливість до змін ключа: навіть мінімальна модифікація секретного ключа призводить до повністю відмінного результату шифрування, що є важливим фактором захисту від атак, орієнтованих на підбір ключа або виявлення залежностей між ключем і шифротекстом.

Для оцінювання криптографічної якості псевдовипадкових бітових послідовностей, згенерованих алгоритмом WBC2, було проведено тестування за допомогою NIST Statistical Test Suite (STS) з використанням набору даних обсягом 1 048 576 байт (8 Мбіт). Результати перевірки, наведені в таблиці, засвідчили, що алгоритм успішно проходить усі основні тести на випадковість. Зокрема, у тестах на частоту появи бітів ("Frequency" та "Block Frequency") зафіксовано збалансоване співвідношення між кількістю нулів і одиниць у кожній послідовності, відхилення не перевищують 0,07 % від загального обсягу, що повністю відповідає допустимим межам за рівня значущості $\alpha = 0{,}01$. Це свідчить про відсутність статистичного зміщення в розподілі бітів, тобто про забезпечення базової рівноваги сигналу.

ТАБЛИЦЯ. Статистичне тестування WBC2 за методикою NIST

Назва тесту	Кількість проходжень	P-value	Висновок
Frequency	10/10	0.350	Успішно
BlockFrequency	10/10	0.911	Успішно
Cumulative Sums (двічі)	10/10	0.213 – 0.740	Успішно
Runs	10/10	0.350	Успішно
Longest Run	10/10	0.213	Успішно
Rank	10/10	0.740	Успішно
FFT	10/10	0.534	Успішно
NonOverlapping Template	113 тестів (деякі 9/10)	0.017 – 0.911	Успішно
Overlapping Template	10/10	0.740	Успішно
Universal	10/10	0.017	Успішно
Approximate Entropy	8/10	0.213	Успішно (допустиме відхилення)
Random Excursions	7/7 (кілька спроб)	0.04 – 0.86	Успішно
Random Excursions Variant	7/7	0.01 – 0.93	Успішно
Serial	10/10	0.534 – 0.991	Успішно
Linear Complexity	10/10	0.213	Успішно

Загалом результати тестування демонструють, що алгоритм WBC2 (Rubikrypt) формує криптографічно якісні псевдовипадкові послідовності, які не містять статистично значущих відхилень, структурних або предикативних елементів. Алгоритм повністю відповідає вимогам до генераторів випадкових бітових потоків, що застосовуються у сфері симетричної криптографії, генерації ключів, а також у інших критично важливих галузях інформаційної безпеки.

Таким чином, проведене тестування підтверджує відповідність алгоритму WBC2 сучасним криптографічним стандартам з точки зору статистичної випадковості вихідних даних.

Висновки. У даній роботі запропоновано вдосконалену модифікацію раніше запропонованого алгоритму WBC1, здійснено аналіз складності й ефективності нового симетричного блочного алгоритму WBC2. Проведене тестування алгоритму, зокрема, за методиками NIST STS, підтвердило високу якість генерації криптографічних послідовностей, лавинний ефект та статистичну стійкість алгоритму. Результати демонструють, що WBC2 забезпечує рівномірний розподіл бітів, низьку кореляцію між відкритими та зашифрованими даними, а також стійкість до диференціальних і лінійних атак. Це підтверджує практичну придатність алгоритму для використання в інформаційних системах із підвищеними вимогами до захисту інформації.

Одним із напрямів підвищення ефективності алгоритмів є використання паралельних обчислень, що значно скорочує час виконання без втрати безпеки. Подальші дослідження спрямовуються на розроблення паралельної модифікації PWBC2 та квантової версії WBCQ, яка використовує параметризоване квантове змішування (PQM) для підвищення динамічності ключа і нелінійності перетворень. У перспективі передбачається інтеграція алгоритмів сімейства WBC у гібридні криптографічні системи, стійкі до атак Гровера, а також проведення розширеного тестування на реальних апаратних платформах і в умовах квантового моделювання.

Розроблений алгоритм WBC2 може розглядатися як спеціалізоване розширення національного криптографічного арсеналу. Якщо «Калина» – це універсальний, високошвидкісний шифр для класичної моделі, то WBC2 спеціалізується на протидію загрозам у моделі «білого ящика», де атакувальник має повний доступ до реалізації. Крім того, WBC2 може бути інтегрований в український криптостек. Наприклад, сеансовий ключ для WBC2 може генеруватися за допомогою «Скелі», а автентифікація шифротексту WBC2 може забезпечуватися «Вершиною» на базі «Купини». WBC2 може слугувати як основний блочний шифр у системах, де загроза аналізу програмного коду є критичною. Таким чином, робота продовжує традицію створення власних математично обґрунтованих криптографічних алгоритмів в Україні, демонструючи життєздатність та інноваційність вітчизняної наукової школи.

Подяка. Автор висловлює щиру вдячність за підтримку та цінні поради академіку НАН України В.К. Задіраці та члену-кореспонденту НАН України А.М. Кудіну.

Фінансування. Автор не отримував фінансування для проведення досліджень та підготовки цієї роботи.

References

1. Zadiraka V.K. Modern methods of solving information security problems. *Visnyk NAN Ukrainy*. 2014. 5. P. 65–69. (in Ukraine)
2. Zadiraka V., Kudin A., Shvidchenko I., Bredelev B. Cryptographic and steganographic protocols for cloud systems. *Computer technologies in information security*. Ternopil: “Kart-blansh”, 2015. P. 9–41.
3. Zadiraka V., Yakymenko I., Kasianchuk M., Ivasyev S. Theoretical and numerical Krestenson’s basis and its application to problems of cryptographic protection and factorization of multidigit numbers. *Computer technologies in information security*. Ternopil: “Kart-blansh”, 2015. P. 216–260. https://doi.org/10.1109/CADSM.2015.7230841
4. Zadiraka V., Smolarz A. Improving performance of two-key cryptography systems. *Computer technologies for information security*. Lublin: Politechnika Lubelska, 2011. P. 90–119.
5. Kudin A.M. Blockchain and crypto currency on the basis of "proof of accuracy”. *Mathematical and computer modeling. Technical sciences*. 2017. 15. P. 104–108. (in Russian)

6. Zadiraka V.K., Kudin A.M. Cloud computing in cryptography and steganography. *Cybernetics and system analysis*. 2013. **49** (4). P. 584–588. https://doi.org/10.1007/s10559-013-9544-x
7. Kudin A.M. Cryptographic transformations of non-Shannon sources of information. *Cybernetics and system analysis*. 2010. **46** (5). P. 813–819. https://doi.org/10.1007/s10559-010-9263-5
8. Gorbenko I.D., Dolhov V.I., Oleinikov R.V. et al. The Prospective Symmetric Block Cipher "Kalyna": Basic Principles and Specifications. *Applied Radio Electronics*. 2007. Vol. 6, No. 2. P. 195–208. (in Ukrainian)
9. Kuznetsov A.A., Ivanenko D.V., Kolovanova E.P. Perspective block cipher «Kalyna» modelling. *Applied Radio Electronics*. 2014. Vol. 13, No 3. P. 201–207.
10. Gorbenko I., Kuznetsov A., Lutsenko M. and Ivanenko D. The research of modern stream ciphers. *4th International Scientific-Practical Conference Problems of Infocommu-nications*. Kharkiv : Science and Technology (PIC S&T). 2017. P. 207–210. https://doi.org/10.1109/INFOCOMMST.2017.8246381
11. DSTU 7624:2014. Information Technology. Cryptographic Protection of Information. Symmetric Block Transformation Algorithm. [Text]. Effective from July 1, 2015. Kyiv: Ministry of Economic Development of Ukraine, 2015. (in Ukrainian)
12. DSTU 7564:2014. Information Technology. Cryptographic Protection of Information. Hash Function. (in Ukrainian) https://usts.kiev.ua/wp-content/uploads/2020/07/dstu-7564-2014.pdf (accessed: 23.10.2025)
13. DSTU 8845:2019. Information Technology. Cryptographic Protection of Information. Symmetric Stream Transformation Algorithm. (in Ukrainian) https://nure.ua/wp-content/uploads/2020/Konkurs/dstu.pdf (accessed: 23.10.2025)
14. DSTU 8961:2019. Information Technology. Cryptographic Protection of Information. Asymmetric Encryption and Key Encapsulation Algorithms. (in Ukrainian) http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=88056 (accessed: 23.10.2025)
15. DSTU ISO/IEC 18033-3:2015 (ISO/IEC 18033-3:2010, IDT). Information Technology. Security Techniques. Encryption Algorithms. Part 3. Block Ciphers. (in Ukrainian)
16. DSTU ISO/IEC 10116:2019 (ISO/IEC 10116:2017, IDT). Information Technology. Security Techniques. Modes of Operation for n-bit Block Ciphers. (in Ukrainian)
17. National Institute of Standards and Technology. NIST Special Publication 800-67: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. January 2012. http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf
18. National Institute of Standards and Technology, "FIPS-197: Advanced Encryption Standard", November 2001. Available at: http://csrc.nist.gov/publications/ fips/fips197/fips-197.pdf/
19. Matsui M. et al. A Description of the Camellia Encryption Algorithm, Internet Engineering Task Force, Request for Comment 3713, April 2004. http://www.ietf.org/rfc/rfc3713.txt 5
20. Gorbenko I. et al. The Kalyna Symmetric Block Cipher – the New National Standard of Ukraine. *Radiotekhnika*. 2015. Iss. 181. P. 5–22. (in Ukrainian) http://nbuv.gov.ua/UJRN/rvmnts_2015_181_3
21. Kuznetsov O.O., Frolenko V.O., Yeromin E.S., Ivanenko D.V. Research of cross-platform implementations of stream symmetric ciphers. *Radioengineering*. 2014. No.193. P. 94–106. https://doi.org/10.30837/rt.2018.2.193.10
22. ISO/IEC 10116:2006. Available at: https://www.iso.org/standard/38761.html (accessed: 23.10.2025)
23. ISO/IEC 10116:2017. Available at: https://www.iso.org/standard/64575.html (accessed: 23.10.2025)
24. NIST Statistical Test Suite: https://csrc.nist.gov/Projects/Random-Bit-Generation/Documentation-and-Software (accessed: 23.10.2025)
25. Baranov I. Symmetric Block Algorithm WBC1 and Analysis of Its Implementation Complexity. *Cybernetics and Computer Technologies*. 2025. 1. P. 64–73. (in Ukrainian) https://doi.org/10.34229/2707-451X.25.1.6
26. Biham E., Shamir A. Differential Cryptanalysis of the Data Encryption Standard. Springer Verlag. 1993. 188 p. https://doi.org/10.1007/978-1-4613-9314-6
27. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. Wiley. 1996. 758 p.
28. Diffie W., Hellman M.E. New Directions in Cryptography. *IEEE Transactions on Information Theory*. 1976. **22** (6). P. 644–654. https://doi.org/10.1109/TIT.1976.1055608
29. Biham E. New Types of Cryptanalytic Attacks Using Related Keys. In: Helleseth, T. (eds) *Advances in Cryptology* – EUROCRYPT '93. EUROCRYPT 1993. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. 1994. Vol. 765. P. 398–409. https://doi.org/10.1007/3-540-48285-7_34
30. Biham E., Shamir A. Differential Cryptanalysis of DES-like Cryptosystems. Advances in Cryptology-CRYPTO' 90. CRYPTO 1990. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. 1991. Vol 537. P. 2–11. https://doi.org/10.1007/3-540-38424-3_1
31. Matsui M. Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (eds) Advances in Cryptology – EUROCRYPT '93. EUROCRYPT 1993. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. 1994. Vol. 765. P. 386–397. https://doi.org/10.1007/3-540-48285-7_33

32. Matsui M.The First Experimental Cryptanalysis of the Data Encryption Standard. *In Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '94)*. Springer-Verlag, Berlin, Heidelberg. 1994. P. 1–11. https://doi.org/10.1007/3-540-48658-5_1
33. Nyberg K. Linear Approximation of Block Ciphers. Advances in Cryptology – EUROCRYPT ’94. Ed. by Alfredo De Santis. Lecture Notes in Computer Science. Springer, 1995. 950. P. 439–444. https://doi.org/10.1007/BFb0053460
34. Kocher P.C. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: *Koblitz N. (eds) Advances in Cryptology – CRYPTO ’96. CRYPTO 1996. Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg. 1996. Vol 1109. https://doi.org/10.1007/3-540-68697-5_9
35. Kocher P., Jaffe J., Jun B. Differential Power Analysis. In: *Wiener M. Advances in Cryptology – CRYPTO’ 99*. CRYPTO 1999. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. 1999. Vol. 1666. P. 388–397. https://doi.org/10.1007/3-540-48405-1_25
36. Brumley D., Boneh D. Remote timing attacks are practical. *Computer Networks*. 2005. **48** (5). P. 701–716. https://doi.org/10.1016/j.comnet.2005.01.010
37. Baranov I.A., Khimich O.M., Nikolaievska O.A. Certificate of Registration of Copyright for a Work issued by the State Service of Intellectual Property of Ukraine: Computer Program “Rubikrypt Software Suite for Data Encryption and Decryption Based on the Symmetric Block Cryptographic Algorithms WBC1 and WBC2”. No.141653; published January 15, 2026. (in Ukrainian)

Received/Одержано 23.10.2025 Accepted/Прийнято 03.03.2026 Published/Надруковано 27.03.2026

Баранов Ігор Анатолійович,
науковий співробітник
Інституту кібернетики імені В.М. Глушкова НАН України, Київ.
https://orcid.org/0000-0002-5500-6210
vlasov@ukr.net

UDC 519.6

Igor Baranov

Cryptographic Complexity Analysis and Testing of the Symmetric Block Algorithm WBC2

V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine, Kyiv
Correspondence: vlasov@ukr.net

Introduction. Modern developments in the field of block-oriented algorithms are aimed at improving performance, reducing computational costs, and integrating such algorithms into quantum-resistant national standards. To enhance the security of the WBC1 algorithm, this paper proposes its modification – the WBC2 algorithm – which is a modern symmetric block cipher that extends the WBC1 model to a more robust, nonlinear, and flexible structure. WBC2 is an interesting example of a cryptographic model with a visual representation (the Rubik’s Cube), which enables the creation of unique key-dependent operations. The paper presents a detailed description of the encryption process, an analysis of the algorithm’s complexity and execution speed, and the results of its testing.

The purpose. The aim of this work is to describe a new symmetric block cryptographic algorithm, WBC2, to investigate its computational complexity and execution speed, and to conduct its testing.

Results. An improved symmetric block cryptographic algorithm, WBC2, has been developed. The complexity analysis and execution speed of the algorithm have been investigated. The applicability of the new algorithm is demonstrated through illustrative examples.

Conclusions. The WBC2 algorithm represents a cryptographically secure encryption method that provides a high level of security through the use of complex dynamic permutations, cyclic shifts, round transformations, extended S-boxes, and diffusion procedures. The increase in processing time in WBC2 is the cost of additional cryptographic complexity and an enhanced avalanche effect. One of the directions for improving the efficiency of the algorithm is the use of parallel computations, which significantly reduces execution time without compromising security. Further research is aimed at developing a parallel modification, PWBC2, and a quantum version, WBCQ, which employs parameterized quantum mixing (PQM) to increase key dynamism and the nonlinearity of transformations.

Keywords: symmetric block cryptographic algorithm, complexity analysis of symmetric block ciphers, performance analysis of cryptographic algorithms, S-box, diffusion, round key, NIST Statistical Test Suite.