АНОТАЦІЯ

Оцінка ефективності алгоритмів блоково-симетричного шифрування на основі використання міні-версій // Дипломна робота ОР «Магістр» // Піх Василь Володимирович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2020 // с., 10 рис., 15 табл., 51 джерело, 3 додатка.

Ключові слова: МІНІ-ВЕРСІЯ, БЛОЧНИЙ СИМЕТРИЧНИЙ ШИФР, КРИПТОАНАЛІЗ, ДИФЕРЕНЦІАЛЬНИЙ КРИПТОАНАЛІЗ, ЛІНІЙНИЙ КРИПТОАНАЛІЗ, АТАКА.

Дана магістерська кваліфікаційна робота присвячена дослідженню методів підвищення ефективності використання міні-версій БСШ для оцінки криптостійкості повних шифрів.

Об'єктом дослідження є процес забезпечення адекватності використання міні-версій щодо оцінки криптостійкості повних шифрів.

Предметом дослідження є оцінка ефективності алгоритмів блоково-симетричного шифрування на основі використання міні-версій.

Алгоритми реалізовані на основі програмного забезпечення, розробленого в середовищі Microsoft Visual Studio 2013 мовою C# та C++.

При дослідженні алгоритмів конкурсантів використовувалася оцінка трьох показників: стійкості алгоритму до відомих криптоаналітичних атак, продуктивність програмної реалізації алгоритму на сучасних персональних комп'ютерах і "статистична безпека" (формування унікальних псевдовипадкових послідовностей).

У результаті проведений порівняльний аналіз міні-версій блочних симетричних шифрів України, та оцінено адекватність їх використання.

Методами розробки обрано:

При аналізі методів і алгоритмів симетричної криптографії використовуються міні-версії шифрів –конкурсантів на алгоритм симетричного шифрування в Україні. При досліджені рівня їх стійкості використовуються основні положення теорій захисту та криптоаналізу, а саме методи лінійного та диференціального аналізу.

При дослідженні стійкості алгоритму були використані методи теорії ймовірності та математичної статистики.

У результаті роботи проведено дослідження можливості використання міні-версій алгоритмів традиційного шифрування щодо отримання їх властивостей повних шифрів за рахунок оцінки стійкості до методів лінійного та диференційного аналізу.

ANNOTATION

Mini-versions-based assessment of block-symmetric encryption algorithms efficiency // Thesis of "Master" Degree// Pikh Vasyl Volodymyrovych// Ternopil National Technical University named after Ivan Pulyuy, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, SBM group -61 // Ternopil, 2020 // P. ., 10 figs., 15 tables, 51 source, 3 appendices.

Keywords: MINI-VERSION, BLOCK SYMMETRIC CIPHERS, CRYPTANALYSIS, DIFFERENTIAL CRYPTANALYSIS, LINEAR CRYPTANALYSIS, ATTACK.

This master's thesis is devoted to the study of methods to improve the efficiency of using mini-versions of block symmetric ciphers (BSC) to assess the cryptographic strength of complete ciphers.

The object of the study is the process of ensuring the adequacy of the use of mini-versions in assessing the cryptographic strength of complete ciphers.

The subject of the study is to evaluate the effectiveness of block-symmetric encryption algorithms based on the use of mini-versions.

The algorithms are implemented on the basis of software developed in Microsoft Visual Studio 2013 in C # and C ++.

In the study of the algorithms of the contestants used the assessment of three indicators: the resistance of the algorithm to known cryptanalytic attacks, the performance of software implementation of the algorithm on modern personal computers and "statistical security" (formation of unique pseudo-random sequences).

As a result, a comparative analysis of mini-versions of block symmetric ciphers of Ukraine was conducted, and the adequacy of their use was assessed.

Development methods selected:

In the analysis of methods and algorithms of symmetric cryptography, mini-versions of ciphers are used – a competition for the symmetric encryption algorithm in

Ukraine. In studying the level of their stability, the main provisions of the theories of protection and cryptanalysis are used, namely the methods of linear and differential analysis.

Methods of probability theory and mathematical statistics were used to study the stability of the algorithm.

As a result, the possibility of using mini-versions of traditional encryption algorithms to obtain their properties of complete ciphers by assessing the resistance to the methods of linear and differential analysis.

ЗМІСТ

ВСТУП

З розвитком науки і техніки постійно з’являються нові вимоги до надійності систем захисту обчислювальної інформації. Основним і найбільш ефективним механізмом криптографічного захисту в таких системах є методи блочно-симетричного шифрування даних (БСШ) [7, 9, 11, 22, 24].

Під ефективністю проектування БСШ розуміється комплексна оцінка алгоритму БСШ, що відображає обґрунтованість і оптимальність вибраних конструкцій для вирішення завдання проектування БСШ, тобто мінімізації апаратних “витрат”, необхідних для забезпечення стійкості алгоритму до атак криптоаналізу [8, 17, 24, 38, 39]. Отже, алгоритм може вважатися ефективним, якщо рівень захисту від відомих криптоаналітичних атак може бути досягнутий ціною істотно менших апаратних витрат.

Дослідження [1, 7, 9, 24, 25, 41], що використовуються в сучасних системах захисту стандартів БСШ показали, що більшість найбільш поширених алгоритмів уже застаріли та не забезпечують необхідну криптостійкість і продуктивність, в той час, коли нові алгоритми вимагають ретельного дослідження та стандартизації. В системах захисту внутрішньо-платіжних банківських систем, комунікаційних і комп’ютерних системах держустанов досі застосовують застарілі стандарти.

Одним із підходів з вивчення та аналізу алгоритмів БСШ є розробка і дослідження зменшених моделей шифрів. Під міні-версією розуміється шифр, який при збереженні математичної структури основних перетворень має менші, ніж шифр-оригінал довжини блоків даних і ключів [7, 9, 11, 29]. Це досягається пропорційним зменшенням відповідних довжин блоків даних і ключів вихідного шифру.

Ціллю розробки дипломної роботи являється аналіз основних вимог і характеристик проектування, що пред’являються до алгоритмів-претендентів блочно-симетричного шифрування на роль національного стандарту, методики оцінки характеристик БСШ на основі аналізу ефективності їх проектування.

Також будуть проаналізовані алгоритми БСШ, дослідження ефективності їх міні-версій до лінійного та диференціального криптоаналізу, як залежно від числа раундів перетворення, так і залежно від числа операцій і необхідних витрат пам’яті, характеристики розглянутих криптоалгоритмів.

Об’єктом дослідження є процес забезпечення адекватності використання міні-версій щодо оцінки криптостійкості повних шифрів.

Предметом є оцінка ефективності алгоритмів блочно-симетричних шифрів на основі використання міні-версій.

Задачі: аналіз основних вимог щодо побудови БСШ; дослідження основних методів оцінки криптостійкості БСШ; дослідження побудови міні-версій щодо оцінки криптостійкості повних шифрів; оцінка адекватності використання міні-версій БСШ для оцінки ефективності повних шифрів.

Теоретична значущість одержаних результатів полягає у підтверджені можливості використання міні-версій алгоритмів БСШ щодо їх використання при оцінці ефективності БСШ на основі порівняння результатів оцінки міні-версій як теоретичних, так і практичних за допомогою програмного макету, та визначені вимог щодо формування міні-версій БСШ.

Практичне значення результатів магістерських досліджень полягає у наступному: надання пропозицій щодо формування S-box-ів для міні-версій БСШ; проведені порівняльні оцінки БСШ на основі диференційного та лінійного аналізу.

Апробація результатів магістерської роботи. Окремі результати роботи доповідались на VIII науково-технічній конференції «Інформаційні моделі, системи та технології», Тернопіль, ТНТУ, 9 – 10 грудня 2020 р.

РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ ОЦІНКИ ЕФЕКТИВНОСТІ БЛОКОВО-СИМЕТРИЧНИХ ШИФРІВ. ОСНОВНІ ВИМОГИ ЩОДО ПОБУДОВИ БСШ

1.1 Основні вимоги щодо побудови БСШ

На сьогоднішній час проводиться аналіз основних вимог щодо побудови блочно-симетричних шифрів [17].

Найбільш поширеним підходом попереднього аналізу сучасних криптоалгоритмів в цілому, та блочно симетричних шифрів (БСШ) зокрема, є оцінка трьох показників [18, 20, 25]: стійкості алгоритму до відомих криптоаналітичних атак, продуктивність програмної реалізації алгоритму на сучасних комп'ютерах і "статистична безпека" (формування унікальних псевдовипадкових послідовностей). Однак даний підхід являється недостатнім, якщо необхідне створення або вибір алгоритму, що претендує на роль національного стандарту, а значить що припускає його масове впровадження в різних сферах застосування. В такому випадку, третім за значущістю критерієм, після криптостійкості і продуктивності, стає "вартість" реалізації алгоритму на різних програмно-апаратних платформах [12, 16].

Під ефективністю проектування БСШ розуміється комплексна оцінка алгоритму БСШ, що відображає обґрунтованість і оптимальність вибраних авторами конструкцій для вирішення задачі проектування БСШ, тобто мінімізації апаратних "витрат", необхідних для забезпечення стійкості алгоритму до відомих атак криптоаналізу [7].

Отже, алгоритм може вважатися ефективним, якщо досягається рівень захисту від відомих криптоаналітичних атак не може бути, досягнутий ціною істотно менших апаратних витрат.

Тому метою методики оцінки БСШ є формування порівняльної оцінки ефективності проектування алгоритмів-претендентів [27].

Таблиця 1.1 – Основні апаратні вимоги до алгоритмів конкурсу БСШ

Критерії	Конкурс AES	NESSIE	Національний конкурс України
Довжина блока	128 біт	64, 128, 256 біт	128, 256, 512 біт
Довжина ключа	128, 192,256 біт	128, 256, 512 біт	128, 256,512 біт
Процесор	платформа 32х	платформа 32х	платформа 32х і 64х
Клас безпеки	задовільний	нормальний	високий
RAM	64 Мб	4 Кб	8 Кб
ROM	100 Кб	250 Кб	40 Кб

Аналіз результатів та практичних рекомендацій, що отримані та сформульовані в ході виконання проектів AES та NESSIE, дозволив зробити висновок, що на цей час не існує єдиної теорії проектування обчислювально стійких БСШ. Міжнародні криптографічні конкурси показали, що основну увагу до стандартів БСШ визначають до її криптостійкості, можливості використання різних довжин ключів (від 128 до 512 біт), а також їх апаратна ємкість та можливість побудови на різних платформах.

1.2 Класифікація атак на БСШ

При розробці будь-яких криптографічних схем, їх стійкість визначається, перш за все, стійкістю до відомих криптоаналітичних атак, які направлені на виявлення недоліків.

Спроба криптоаналізу називається атакою. Перш ніж класифікувати атаки, введемо ряд позначень: відкритий текст будемо позначати буквою *x*, шифртекст – буквою *y* (як *x* може виступати будь-яка послідовність бітів: текстовий файл, оцифрований звук, точковий малюнок і т.д.). Нехай для шифрування і розшифрування використовуються ключі *k* і *k'* відповідно; позначимо функцію шифрування *Ek*, розшифрування – *Dk'*. Тоді виконуються співвідношення $Ek(x) = y$, $Dk'(y) = x$.

Відомі чотири основні типи криптоаналітичних атак. У кожному разі

передбачається, що криптоаналітик знає використовуваний алгоритм шифрування. Класифікація основних видів криптоаналітичних атак наведена на рис. 1.1.

Рисунок 1.1 – Криптоаналітичні атаки на БСШ

Атаки можна також класифікувати за обсягом ресурсів, необхідних для їх здійснення:

Пам’ять – обсяг пам’яті, необхідний для реалізації атаки;

час – кількість елементарних операцій, які необхідно виконати;

дані – необхідний обсяг відкритих і відповідним їм зашифрованих текстів.

У деяких випадках ці параметри є взаємозалежними: наприклад, за рахунок збільшення пам’яті можна скоротити час атаки.

Здатність криптосистеми протистояти атакам криптоаналітика називається стійкістю [4]. Кількісно стійкість вимірюється як складність найкращого алгоритму, що приводить криптоаналітика до успіху з прийнятною вірогідністю. Універсальний метод прямого перебору безлічі всіх можливих ключів дозволяє отримати оцінку зверху для стійкості алгоритму шифрування [13]. Проблема всієї сучасної криптографії – це відсутність нижньої межі стійкості; довжина ключа задає лише загальний обсяг простору ключів, але завжди є ймовірність вгадати рішення.

Розрізняють стійкість ключа (складність розкриття ключа найкращим відомим алгоритмом), стійкість безключового читання, і ймовірність нав'язування неправдивої інформації. Аналогічно можна розрізняти стійкість власне криптоалгоритму, стійкість протоколу, стійкість алгоритму генерації та поширення ключів [7].

Залежно від складності злому алгоритми забезпечують різні ступені захисту. На перше місце ставиться принципова можливість отримання з перехоплення деякої інформації про відкритий текст або використаний ключ. Існують безумовно стійкі (або теоретично стійкі), доказовою стійкі і імовірно стійкі криптоалгоритми.

Криптоаналіз використовується на сьогоднішній день у великій кількості різних областей. Криптоаналіз ставить своїм завданням в різних умовах отримати додаткові відомості про ключі шифрування, щоб значно зменшити діапазон ймовірних ключів. При цьому, результати можуть варіюватися за ступенем практичної застосовності.

Перевагами криптоаналізу є: простота реалізації та висока швидкість перетворення; недолік – відсутність математичного перетворення криптостійкості.

Основними методами отримання інформації є лінійний і диференційний криптоаналіз [6, 14, 20, 21]. Розглянемо їх більш детально.

1.3 Лінійний криптоаналіз

Лінійний криптоаналіз відносять до атак на основі "відомих відкритих текстів". Вперше ця атака була запропонована Мацуї для криптоаналізу FEAL [9], а потім для DES [2], пізніше атака на DES була вдосконалена Біхамом [11]. Техніка лінійного криптоаналізу була розширена Ніберг [9], яка запропонувала концепцію лінійних шаблонів, яка полягає у використанні безлічі різних лінійних характеристик ("активізуючих" різні множини S блоків), які задовольняють деякому вхідному і вихідному шаблону.

Цей метод криптоаналізу заснований на пошуку і використанні найбільш “правдоподібних” лінійних (афінних) співвідношень для апроксимації нелінійних перетворень. Лінійна характеристика, яка визначає вид апроксимуючої лінійної функції, записується як

$$X \cdot \Lambda_X \oplus Y \cdot \Lambda_Y \oplus K \cdot \Lambda_K = \nu, \tag{1.1}$$

де *X, Y, K* – відповідно вектори входу, виходу і ключа; Λ_X, Λ_Y, Λ_K – відповідно шаблони (“маски”) входу, виходи і ключа; ν – найбільш ймовірне значення апроксимуючої функції (0 або 1); точкою (a • b) позначено скалярне множення векторів *a* і *b*.

Характеристика може бути застосована для криптоаналізу якщо функція $\nu_{\Lambda_X,\Lambda_Y,\Lambda_K,K}(X)$, що апроксимується не збалансована, тобто ймовірність “вірної” апроксимації перевищує 1/2.

Так якщо розподіл ймовірності деякої двійкової випадкової величин *Xi* задано як

$$\Pr(X_i = v) = \begin{cases} p_i & , v = 0 \\ 1 - p_i & , v = 1 \end{cases} \tag{1.2}$$

тоді *pi* може бути виражена через відхилення (ε_i) ймовірності від 1/2 як $p_i = 1/2 + \varepsilon_i$, $-1/2 \le \varepsilon_i \le +1/2$, або через дисбаланс як $(2\varepsilon_i(= (2p_i - 1($.

При побудові багатоциклової лінійної характеристики передбачається, що складові її характеристики незалежні, тому для обчислення ймовірності результуючої характеристики Мацуї використовує принцип накопичувальної суми [11].

Лема “про накопичену суму” [19]:

Для *r* незалежних довічних випадкових величин $X_1, X_2, ..., X_r$ справедливо

$$\Pr(X_1 \oplus \ldots \oplus X_r = 0) = 1/2 + 2^{r-1} \prod_{i=1}^{r} \varepsilon_i \tag{1.3}$$

або еквівалентно

$$\varepsilon_{1,\ldots,r} = 2^{r-1} \prod_{i=1}^{r} \varepsilon_i \tag{1.4}$$

де $\varepsilon_{1,\ldots,r}$ – "відхилення" суми $X_1 \oplus \ldots \oplus X_r = 0$.

З наведеної леми випливає, що, з одного боку, якщо $p_i = 0$ або 1 для всіх i, то $\Pr(X_1 \oplus \ldots \oplus X_r = 0) = 0$ або 1, а з іншого боку, якщо хоча б один $p_i = 1/2$, то $\Pr(X_1 \oplus \ldots \oplus X_r = 0) = 1/2$. Тому, при побудові багатоциклової лінійної характеристики, як було зазначено вище, в неї можуть включатися тільки одноциклові характеристики з $p_i \neq 1/2$.

Мацуї показав, що для успішного застосування атаки лінійного криптоаналізу необхідно близько ε^{-2} відомих відкритих текстів, де ε – відхилення від 1/2 ймовірності виконання обраної лінійної апроксимації, що охоплює весь шифр. Тому алгоритм вважається вразливим до лінійного криптоаналізу, якщо ймовірність деякої суми "вхід-вихід", що охоплює всі, крім декількох циклів (зазвичай 2 або 3) значно більше ніж $2^{-n/2}$.

Узагальнення лінійного криптоаналізу Мацуї було наведено в [6]. Сумою "вхід-вихід" S_i циклу i ітеративного шифру називається сумою за модулем 2 (XOR) двох збалансованих булевих функцій: f_i – від вхідного вектора $x_i = y_{i-1}$ циклу i та g_i – від вихідного вектора y_i циклу i, тобто

$$s_i = f_i(y_{i-1}) \oplus g_i(y_i) \tag{1.5}$$

Суми "вхід-вихід" сусідніх циклів називаються пов'язаними, якщо вихідна функція попереднього циклу збігається з вхідною функцією наступного, тобто $g_i = f_i + 1$. R-циклова сума "вхід-вихід" може бути записана як сума відповідних пов'язаних сум окремих циклів:

$$s^{(r)} = \bigoplus_{i=1}^{r} s_i = f_1(x_1) \oplus g_r(y_r) \tag{1.6}$$

Гомоморфізм з групи $(Z_2^n, \oplus)$ в групу $(Z_2, \oplus)$ називається булевим гомоморфізмом для деякої операції $\oplus$. Булева функція f є таким гомоморфізмом, якщо вона не дорівнює константі нуля і $\forall\, U, V \in Z_2^n$ справедливо $f(U \oplus V) = f(U) \oplus f(V)$. Сума "вхід-вихід" для циклів від i до j $(j \geq i)$ є гомоморфною, якщо вхідна функція є булевим гомоморфізмом для $\oplus i$, а вихідна функція є булевим гомоморфізмом для $j \oplus 1$.

Таким чином, успіх атаки лінійного криптоаналізу *r*-циклового ітеративного шифру залежить від можливості виявлення (*r-1*)-циклової гомоморфною суми “вхід-вихід” зі значним дисбалансом.

Після *R*-1 раунду лінійного наближення, виявленого для шифру *R* раундів з досить великою лінійною ймовірністю зміщення, можна атакувати шифр, відновлюючи біти останнього підключа. Процес включає в себе часткову розшифрування останнього раунду шифру. Зокрема, для всіх можливих значень часткових цільових підключів біти шифртексту зашифровані ексклюзивним АБО з бітами часткових цільових підключів і на виході операції виконуються в зворотному порядку через відповідні S-бокси [6]. Це робиться для всіх відомих зразків відкритих текстів / зразків шифртексту і кількість зберігається для кожного значення часткового цільового підключа.

При цьому, необхідно розуміти основні етапи лінійного криптоаналізу, які наведені на рис. 1.2.

На першому етапі, для проведення лінійного криптоаналізу використовується апроксимація роботи частини шифру з виразом, що є лінійним, де лінійність відноситься до побітової операції mod-2 (тобто, виключним АБО, позначеним “⊕”). Проведені дослідження є логічним продовженням робіт [6]:

Такий вираз має вигляд:

$$X_{i1} \oplus X_{i2} \oplus ... \oplus X_{iu} \oplus Y_{j1} \oplus Y_{j2} \oplus ... \oplus Y_{jv} = K_{z1} \oplus K_{z2} \oplus ... \oplus K_{zn,} \quad (1.7)$$

де *Xi* представляє *i*-й біт входу $X = [X_1, X2, ...]$ і *Yj* представляє *j*-й біт виходу $Y = [Y_1, Y_2, ...]$, K_z представляє *z*-й біт ключа. Це рівняння, що представляє виключне АБО “суму” вхідних бітів і вихідних бітів.

В роботі [10] наведений як швидкий алгоритм обчислення таблиці апроксимацій *TA(f)* для випадкової функції *f*, визначеної у вигляді:

$$Y = f(X) : \{0,1\}^n \rightarrow \{0,1\}^m \quad (1.8)$$

де f – випадкова функція, $X * \lambda X$, де λ – маска.

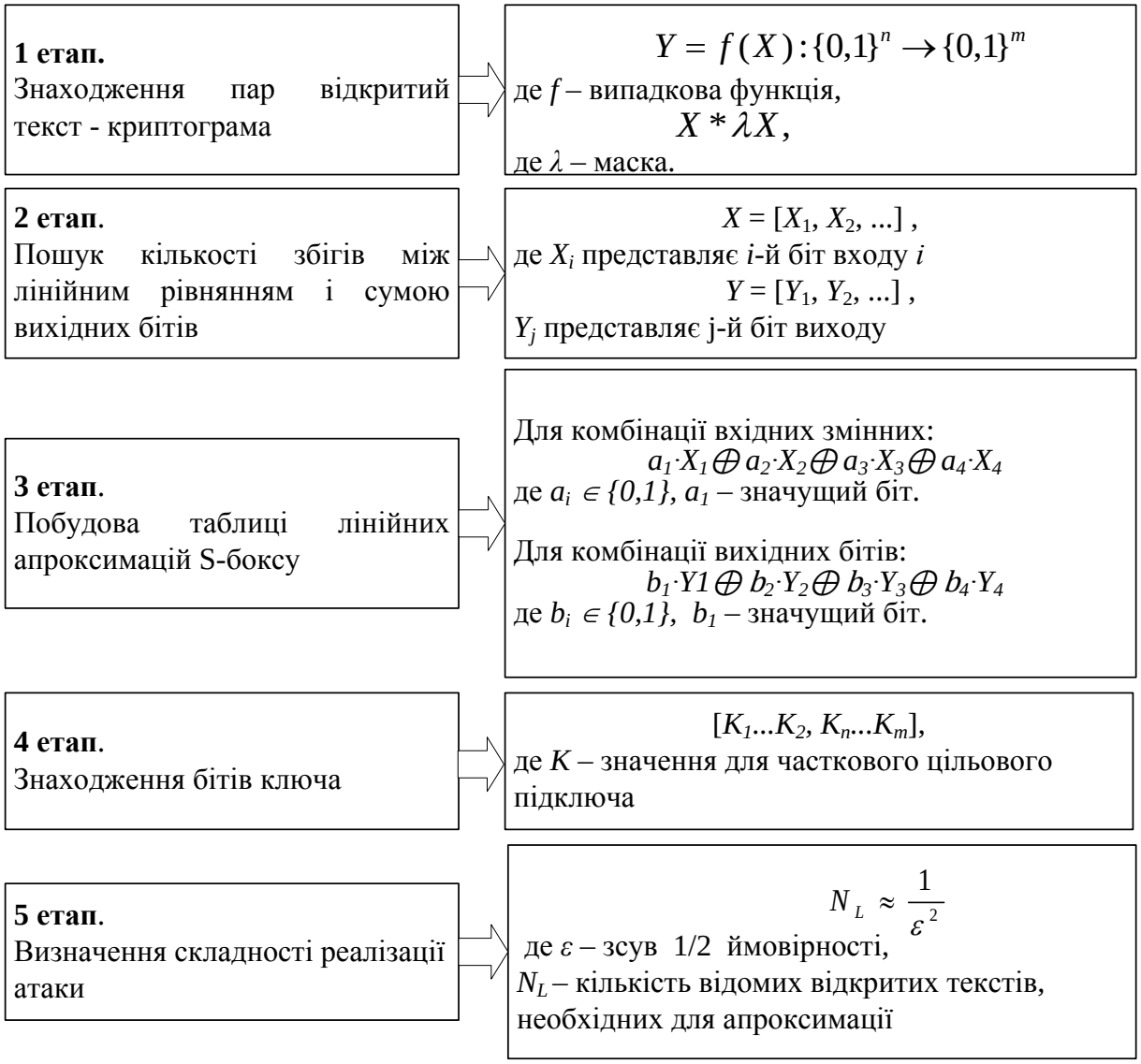


Рисунок 1.2 – Основні етапи лінійного криптоаналізу

Підхід лінійного криптоаналізу – визначення виразів, які мають високу або низьку ймовірність виникнення. Лінійність не слід проводити у всіх вхідних і вихідних значень, інакше шифр буде тривіально слабким. Якщо шифр відображає тенденцію для рівняння (1) проведену з високою ймовірністю або не проведену з великою ймовірністю, це свідчить про бідні здібності рандомізації шифру. Припускаючи, що якщо ми випадково виберемо значення для $X + Y$ біт і розмістимо їх в рівняння вище, ймовірність того, що вираз буде проведено, буде дорівнює 1/2. Це відхилення, або зсув від ймовірності 1/2 для вираження

використовується в лінійному криптоаналізі: чим більше лінійний вираз відрізняється від ймовірності 1/2, тим краще криптоаналітик здатний застосовувати лінійний криптоаналіз [5]. При розробці шифру лінійного наближення, значення фактично представляють собою лінійні наближення S-боксів. Де X_i являє *i*-й біт входу $X = [X_1, X_2, ...]$ і Y_j представляє *j*-й біт виходу $Y = [Y_1, Y_2, ...]$. Це рівняння, що представляє виключне АБО "суму" X вхідних бітів і Y вихідних бітів [4].

На другому етапі здійснюється пошук збігів між вхідними та вихідними бітами, де X – вхідна сума, Y– вихідна сума.

Далі, на третьому етапі будуємо таблицю лінійних апроксимацій. Для лінійної комбінації вхідних змінних, представлених у вигляді

$$a_1 \cdot X_1 \oplus a_2 \cdot X_2 \oplus a_3 \cdot X_3 \oplus a_4 \cdot X_4 \qquad (1.9)$$

де $a_i \in \{0,1\}$ і " $\cdot$ " представляє бінарне I, 16-бітне значення і представляє собою двійкове значення $a_1a_2a_3a_4$, де a_1 значущий біт. Аналогічно, для лінійної комбінації вихідних бітів

$$b_1 \cdot Y_1 \oplus b_2 \cdot Y_2 \oplus b_3 \cdot Y_3 \oplus b_4 \cdot Y_4 \qquad (1.10)$$

де $b_i \in \{0,1\}$, 16-бітне значення представляє бінарний вектор $b_1b_2b_3b_4$.

На четвертому етапі відбувається знаходження бітів ключа, яке виробляється в такий спосіб: після $R–1$ раунду лінійного наближення, виявленого для шифру R раундів з досить великою лінійною ймовірністю зміщення, можна атакувати шифр, відновлюючи біти останнього підключа. Для кожного зразка відкритого тексту / шифртексту, перевіряють всі значення для часткового цільового підключа $[K_1...K_2, K_n...K_m]$. Для кожного значення часткового підключа, ми збільшуємо кількість щоразу, коли ми визначаємо значення $[U_1..U_2, U_n...U_m]$, запустивши дані в зворотному напрямку через частковий цільовий підключ і S-бокси. Кількість, яка відхиляється за величиною від половини кількості зразків відкритого тексту / шифртексту передбачає правильне значення [6].

На п’ятому етапі відбувається визначення складності атак, при цьому ε представляє зміщення 1/2 ймовірності того, що лінійний вираз виконується для повного шифру. В роботі [2] головна ідея полягає в тому, що число відомих відкритих текстів, необхідних в нападі, пропорційно ε^{-2}.

$$N_L \approx \frac{1}{\varepsilon^2} \tag{1.11}$$

Таким чином, лінійний криптоаналіз дозволяє проводити атаки на БСШ. При цьому, принцип роботи полягає у тому, щоб побудувати відношення між відкритим текстом, зашифрованим текстом та ключом. Та використання їх з парами відкритий текст/ шифртекст для отримання бітів ключа.

Іншим не менш важливим методом криптоаналізу на думку вчених є диференціальний криптоаналіз.

1.4 Диференціальний криптоаналіз

Вперше диференціальна атака була запропонована Біхамом і Шаміром криптоаналізу DES [14]. Дана атака використовує нерівномірний розподіл ймовірностей $p(\Delta y \mid \Delta x, k)$ зміни виходу Δy шифратора під дією фіксованої зміни його входу $\Delta x \neq 0$ для відновлення ключа шифрування k, від якого залежить ця ймовірність.

Диференціальний криптоаналіз базується на понятті різниці Δ, що розглядається в контексті групової операції, використовуваної для введення ключа. Припустимо, що циклове перетворення складається з операції “складання” з цикловим ключем ki (скористаємося адитивним позначенням довільної групової операції) і фіксованого нелінійного (щодо операції ключового “складання”) перетворення f:

$$y_i = \mathrm{f}(x_i + k_i) \tag{1.12}$$

Тоді, вхідна і відповідна їй вихідна Δy_i різниці можуть бути задані за допомогою пари вхідних значень (x_i, x_i'):

$$\Delta x_i = x_i' - x_i,\ \ \Delta y_i = y_i' - y_i = \mathrm{f}(x_i + k_i) - \mathrm{f}(x_i' + k_i) \tag{1.13}$$

Фіксоване трансформування вхідної різниці $\Delta x_i = \alpha_i$ в вихідну $\Delta y_i = \beta_i$, з імовірністю $p(\alpha_i \to \beta_i)$ на циклі i процедури шифрування називається одноцикловою характеристикою. Для "транспортування" різниці через цикли шифрування використовується поняття багатоциклової характеристики, яка отримується за допомогою конкатенації сусідніх одноциклових характеристик, що задовольняють співвідношенню $\beta_{i-1} = \alpha_i$, при цьому ймовірність результуючої r-циклової характеристики визначається як добуток ймовірностей складових її одноциклових характеристик:

$$P = \prod_{i=1}^{r} p(\alpha_i \to \beta_i) \tag{1.14}$$

Стосовно до концепції Марківського шифру, в [8] дано узагальнення диференціального криптоаналізу Біхам-Шаміра. Так якщо припустити, що всі циклові підключи формуються незалежно та рівноймовірно, то для такого шифру наступна умовна ймовірність

$$p(\Delta y_i = \beta \mid \Delta x_1 = \alpha,\ x_1 = \gamma) \tag{1.15}$$

не залежить від вхідного значення γ. З цього випливає, що послідовність

$$\Delta x_1, \Delta y_1, \Delta y_2, \ldots, \Delta y_i \tag{1.16}$$

є марківським ланцюгом і має стаціонарний розподіл для ненульових різниць.

Пару $(\Delta x_1, \Delta y_i) = (\alpha, \beta)$, яка визначає вхідну різницю і різницю після циклу i, називають i-цикловим диференціалом, а відповідну послідовність значень $(\alpha, \beta_1, \beta_2, \ldots, \beta_{i-1}, \beta)$ називають i-цикловою характеристикою, що належить i-цикловому диференціалу (α, β). Імовірність i-циклового диференціала $p(\Delta y_i = \beta \mid \Delta x_1 = \alpha)$ визначається як сума ймовірностей належних йому характеристик:

$$p(\Delta y_i = \beta \mid \Delta x_1 = \alpha) = \sum_{\forall \beta_1, \ldots, \beta_{i-1} \neq 0} p(\Delta y_1 = \beta_1, \ldots, \Delta y_{i-1} = \beta_{i-1}, \Delta y_i = \beta \mid \Delta x_1 = \alpha) \tag{1.17}$$

Так як диференційний криптоаналіз відновлює біти ключа на останньому циклі, то для успішного "нападу" на r-цикловий шифр необхідний $(r\text{-}1)$-цикловий диференціал, ймовірність якого значно вище середньої ймовірності інших диференціалів з аналогічним входом α.

Диференціальний криптоаналіз використовує високу ймовірність певних різних появ відкритого тексту і відмінностей в останньому раунді шифру. Для проведення диференціального криптоаналізу використовуються наступні етапи, які наведені на рис. 1.3.

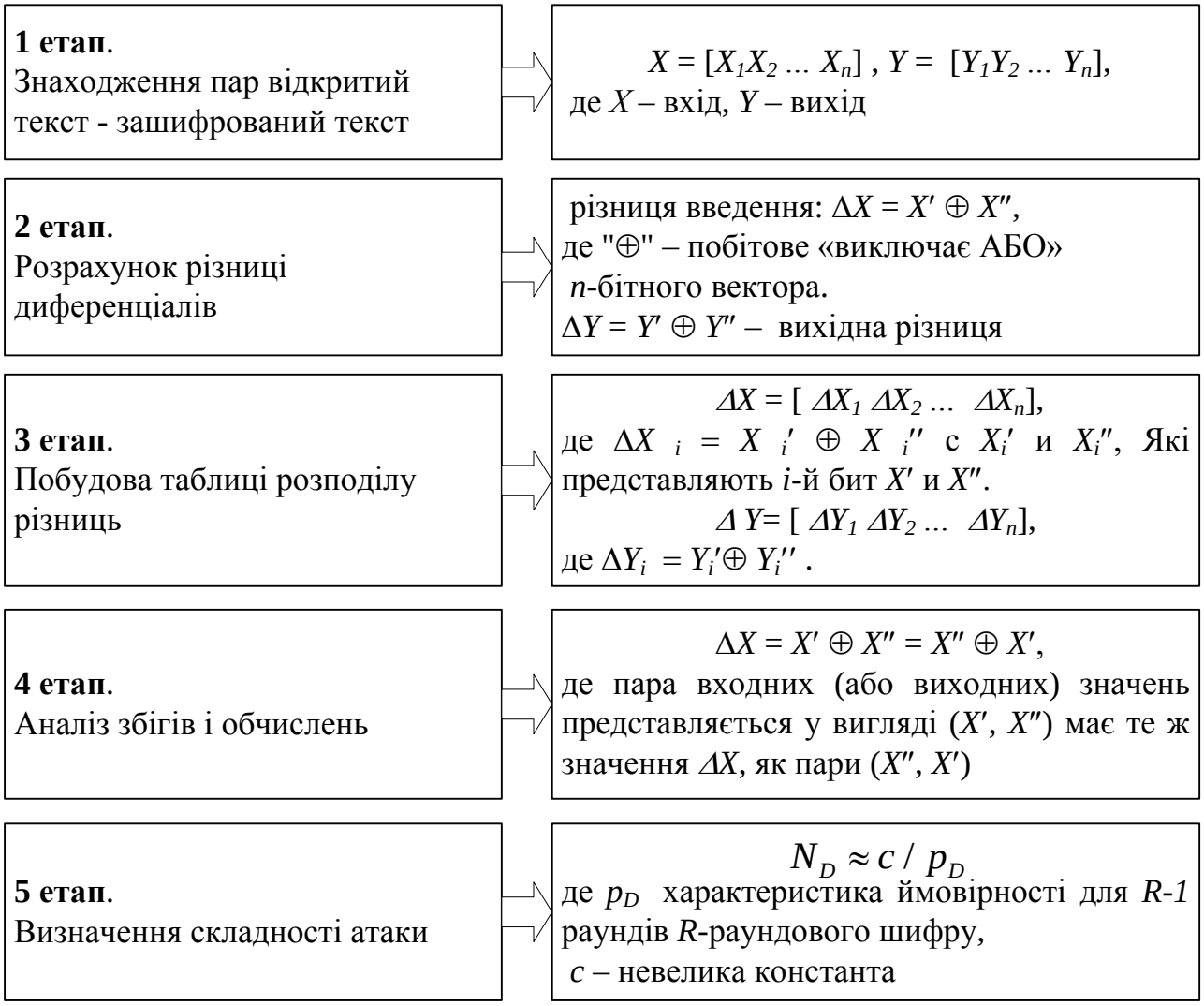


Рисунок 1.3 Основні етапи диференційного криптоаналізу

При цьому, на першому етапі диференціального криптоаналізу проводиться пошук появ відкритого тексту і шифртексту, де вхід $X = [X_1X_2 \dots X_n]$ і вихід $Y = [Y_1Y_2 \dots Y_n]$. Входами в систему є X' і X'' з відповідними виходами Y' і Y''.

Під час другого етапу, визначають різницю диференціалів. При цьому, різниця введення задається $\Delta X = X' \oplus X''$, де “⊕” є побітове виключне АБО n-бітного вектору.

Під час третього етапу потрібно побудувати таблицю розподілу різниць:

$$\Delta X = [\Delta X_1 \Delta X_2 ... \Delta X_n] \tag{1.18}$$

де $\Delta X_i = X_i' \oplus X_i''$ с X_i' і X_i'', які представляють i-й біт X' і X'', відповідно. Аналогічно, $\Delta Y = Y' \oplus Y''$ являє собою вихідну різницю і

$$\Delta Y = [\Delta Y_1 \Delta Y_2 ... \Delta Y_n] \tag{1.19}$$

де $\Delta Y_i = Y_i' \oplus Y_i''$.

Як і з лінійним криптоаналізом, щоб побудувати диференціальні характеристики, ми розглянемо властивості окремих S-боксів і використаємо ці властивості, щоб визначити повну диференціальну характеристику. Зокрема, ми розглянемо відмінності вхідного і вихідного S-боксу для того, щоб визначити різницю пар з високою ймовірністю [1]. Поєднуючи різниці пар S-боксів від раунду до раунду так, що ненульова різниця вихідних бітів з одного раунду відповідають ненульовий різниці вхідних бітів в наступному раунді, що дозволяє знайти високу ймовірність диференціала, що складається з різниць відкритого тексту і різниць на вході в останньому раунді. Біти підключа шифру в кінцевому підсумку обчислюються з різниці виразу, тому що вони беруть участь в обох наборах даних і, отже, з урахуванням їх впливу на різницю включають біти підключа виключне АБО, результатом якого є нуль.

На четвертому етапі проводиться аналіз збігів. Крім того, також розраховують всі значення елементів: пара вхідних (або вихідних) значень представляється у вигляді (X', X'') має те саме значення ΔX, як пари (X'', X'), так як $\Delta X = X' \oplus X'' = X'' \oplus X'$. Крім того, вхідна різниця $\Delta X = 0$ повинна привести до вихідної різниці $\Delta Y = 0$ для відображення один-до-одного з S-боксу.

На п’ятому етапі необхідно визначити складність атак. В цілому, це дуже складно точно визначити кількість обраних пар відкритих текстів, необхідних

для атаки. Проте, це може бути доведено, що існує правило для числа пар обраних відкритих текстів, N_D, необхідних, щоб відрізнити правильні пари

$$N_D \approx c / p_D \tag{1.20}$$

де p_D – диференціальна характеристика ймовірності для $R–1$ раундів R-раундового шифру і c – це невелика константа. Якщо припустити, що випадки різницевих пар в кожному активному S-боксі є незалежними, диференціал характеристики ймовірності задається

$$p_D = \prod_{i=1}^{\gamma} \beta_i \tag{1.21}$$

де кількість активних S-боксів представлено γ і поява особливої різниці пар в i-му активному S-боксі характеристики має ймовірність, представлену β_i.

Як і у лінійному криптоаналізі, необхідно проявляти обережність при "доказі" стійкості в диференціальному криптоаналізу. Обчислення ймовірності диференціальної характеристики ґрунтується на незалежності S-боксів, які беруть участь в наближенні і в реальному шифрі, де існує залежність між даними, які надходять в різні S-бокси.

Таким чином, при проведенні диференціального криптоаналізу, найбільш трудомістким етапом є побудова таблиць різності диференціалів на основі пар відкритий текст – криптограма, що суттєво впливає на результат криптоаналізу. На сьогоднішній день важливим питанням є оцінка блочно-симетричних шифрів, яка розглядатиметься у наступному підрозділі.

1.5 Методика оцінки БСШ

В роботах [25, 31, 32] вченими розглядається новий підхід до оцінки показників доказової стійкості блокових симетричних шифрів до атак

диференціального і лінійного криптоаналізу, який ґрунтується на результатах вивчення властивостей шифрів як випадкових підстановок.

Отже, для подолання обчислювальних труднощів [15, 17, 23], властивих аналізу показників стійкості великих шифрів, розвивається методика, яка будується на результатах вивчення властивостей зменшених версій великих прототипів. Відповідно до цієї методики максимуми повних диференціалів і лінійних корпусів шифрів можуть бути отримані розрахунковим шляхом з формул, виведених для випадкових підстановок.

Для подолання труднощів аналізу повномасштабних моделей (алгоритмів) шифрування при дослідженні зменшених моделей прототипів, для них наявних обчислювальних ресурсів виявляється вже цілком достатньо. Вдається в багатьох випадках побудувати зменшені моделі, які зберігають (з урахуванням масштабування) всі властивості своїх прототипів і дозволяють вирішити багато завдань аналізу і порівняння з показниками стійкості великих версій шифрів.

Нижче наведено математичне обґрунтування, що підтверджується численними експериментами з малими шифрами. Закон розподілу ймовірностей $DP^{f_{r+1}}(\Delta x, \Delta z)$ для r + 1 циклу перетворень, де Δz є вихідною різницею *r + 1*-го циклового перетворення. І тоді диференціальна ймовірність $DP^{f_{r+1}}(\Delta x, \Delta z)$ для r + 1-го циклового перетворення може бути визначена зі спільної вірогідності $DP^{f_{r+1}}(\Delta x, \Delta y, \Delta z)$ шляхом її усереднення по безлічі проміжних значень $\Delta y \in Z_2^n$, при цьому

$$DP^{f_{r+1}}(\Delta x, \Delta z) = \sum_{\Delta y \in Z_{2^n}} DP^{f_r}(\Delta x, \Delta y) DP^{f_1}(\Delta z / \Delta x, \Delta y) \qquad (1.22)$$

В результаті утримуємо:

$$DP^{f_{r+1}}(\Delta x, \Delta z) = DP^{f_r}(\Delta x, \Delta y) \to DP^{f_{r+1}}(\Delta x \to \Delta z) = DP^{f_r}(\Delta x \to \Delta y), \qquad (1.23)$$

де $$DP^{f_r}(\Delta x \to \Delta y) = \Pr(\Lambda_f(\Delta x, \Delta y) = 2k). \qquad (1.24)$$

Це означає, що додаткові циклові перетворення вже не змінюють закону розподілу різниць на виході шифру.

Для *R*-циклового шифру SPN структури формування результуючого закону розподілу ймовірностей переходів таблиці повних диференціалів зводиться до послідовного виконання *R* однотипних (одноциклових) перетворень (*R* ітерацій).

Сучасні блокові симетричні шифри при повному наборі шифруючих багатоциклових перетворень мають властивості випадкових підстановок, тобто для них справедливі закони розподілу ймовірностей для комбінаторних показників (інверсій, зростань і циклів), а також закони розподілу ймовірностей повних диференціалів і лінійних апроксимацій, властиві випадковим підстановки відповідного ступеня.

В роботах [8, 13, 24, 41] основним питанням є вибір коефіцієнту та вибір міні S-боксу. В роботах [1, 7] пропонується використання S-боксів для оцінки алгоритмів-конкурсантів.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ АЛГОРИТМІВ БСШ НА ОСНОВІ МІНІ-ВЕРСІЙ

2.1 Аналіз можливості використання міні-версій алгоритмів БСШ

На сьогоднішній день актуальною задачею є оцінка можливості використання міні-версій. Для проведення можливості використання на основі лінійного та диференціального криптоаналізу БСШ багатьма вченими використовуються міні-версії, які дозволяють провести повний лінійний та диференціальний криптоаналіз і скоротити час його проведення. Для проведення досліджень використовуються алгоритми-конкурсанти БСШ конкурсу України. Підхід використання міні-версій для оцінки криптостійкості є актуальні і описаний в роботах [1, 11, 12].

В рамках проведених досліджень розглядалися криптографічні властивості Фейстель-подібних і SPN (Substitution-Permutation Network) блокових шифрів зі зменшеним розміром блоку і ключа (16 або 32 бітів). Доцільність розгляду саме зменшених моделей шифрів пояснюється тим, що для вивчення стійкості шифру до диференціальної і лінійної атаки слід оцінювати ймовірності повних диференціалів і ймовірності лінійних апроксимацій – параметри, які можна оцінити тільки для шифру з невеликим розміром блоку.

У даній дипломній роботі описується оцінка ефективності використання міні-версій БСШ по мінімальному числу виконуваних операцій криптоалгоритму, щоб вийти на показник середніх значень максимумів. Щоб використовувати міні-версії, експерту необхідно визначитися з правильним вибором коефіцієнта зменшення операцій. Коефіцієнт скорочення або мінімізації повинен бути строго обґрунтований. Важливу роль визначає S-box. При використанні S-box підтверджується, що міні-версії можуть бути використані для оцінки криптостійкості того чи іншого шифру для міні-версій. Для правильного проведення диференціального і лінійного криптоаналізу на основі міні-версій, необхідно опрацьовувати S-box. Розглядаються 32-х розрядні міні-версії шифрів з 32-х розрядним ключем. Отже, загальне число різних

підстановок дорівнює потужності ключового простору і для кожного ключа таблиця диференціалів має розмір 2^{32} x 2^{32}.

Для дослідження диференціальних властивостей шифрів використані міні-версії з різними S-боксами. Далі в цій роботі будуть викладені результати проведення аналізу алгоритмів блоково-симетричних шифрів на основі різних структур міні-версій. Для проведення досліджень вводяться коефіцієнти зменшення, що описано в роботах [7, 11, 29, 37].

Проведений аналіз [1,11] показав, що основними вимогами до міні-версій є:

1. Адекватність зменшення математичних операцій в приведених шифрах;

2. необхідні витрати пам'яті щодо проведення оцінки ефективності шифру;

3. кількість раундів перетворення щодо виходу на статистичну безпеку шифру;

4. правильний підбір коефіцієнту зменшення, який не змінює алгебраїчну структуру алгоритму;

5. експертна оцінка використання констант (S-box, векторів ініціалізації).

Для проведення досліджень були взяті міні-версії алгоритмів-конкурсантів. Опис міні-версій деяких шифрів наводяться в роботах [8, 12,18, 29].

У табл. 2.1 показані результати для 16 та 32-бітних міні-версій шифрів.

Таблиця 2.1 – Обчислювальні витрати 16 і 32-бітних міні-версій шифрів

Алгоритм	16-ти бітна міні-версія				32-х бітна міні-версія			
	Стійкість до диференційного криптоаналізу		Стійкість до лінійного криптоаналізу		Стійкість до диференційного криптоаналізу		Стійкість до лінійного криптоаналізу	
	S-box Хейса	Зменшений S-box	S-box Хейса	Зменшений S-box	S-box Хейса	Зменшений S-box	S-box Хейса	Зменшений S-box
Лабіринт	240	229	243	232	243	235	247	235
Калина	233	223	235	226	237	230	238	228
Мухомор	228	222	231	225	233	230	235	227
AES	244	232	250	234	248	237	255	240
ADE	237	225	246	229	243	230	252	232

У табл. 2.1 представлені оцінки стійкості до диференціального і лінійного криптоаналізу необхідних обчислювальних ресурсів (кількість елементарних операцій) для кожної 16 і 32-бітної міні-версії алгоритмів БСШ при використанні S-боксу для шифру Хейса і зменшених версій S-боксів відповідних алгоритмів, тобто зменшених на певний коефіцієнт. Згідно з експертними оцінками наведені табличні уявлення відібраних для проведення експериментів S-боксів для шифру Хейса формуються наступним чином [1].

Провівши аналіз, можна зробити висновок, що 32 і 16-бітні міні-версії, які використовували S-box для шифру Хейса стійкіші до диференціального і лінійного криптоаналізу, ніж S-бокси, зменшені на коефіцієнт.

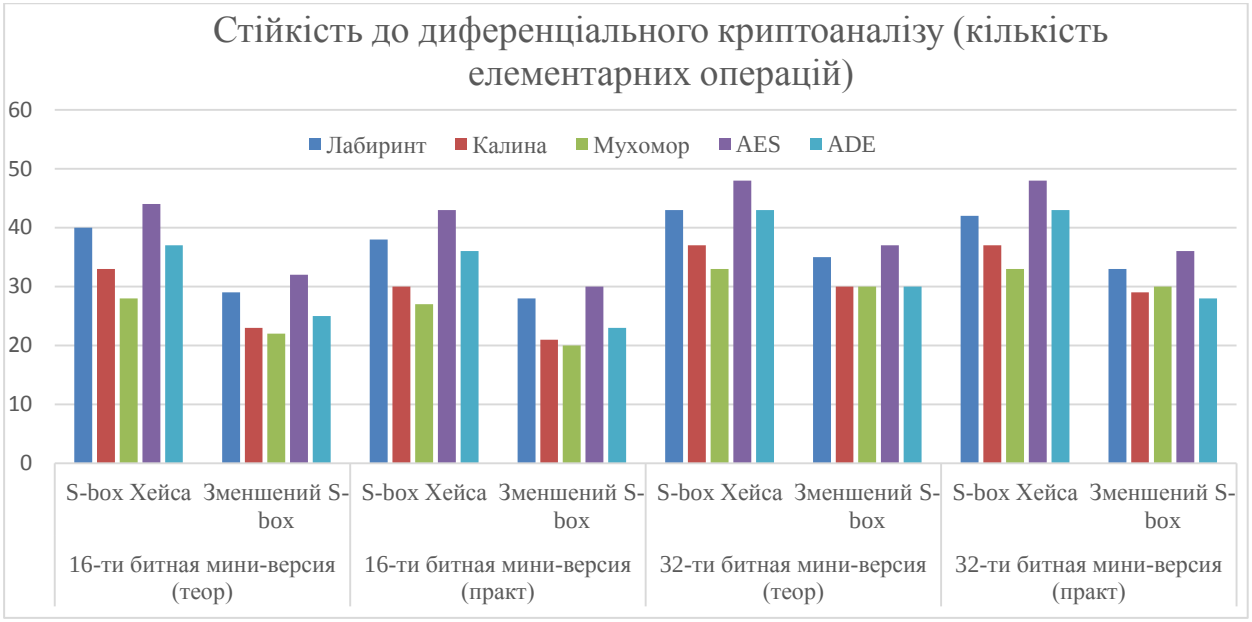


Рисунок 2.1 – Стійкість до диференційного криптоаналізу

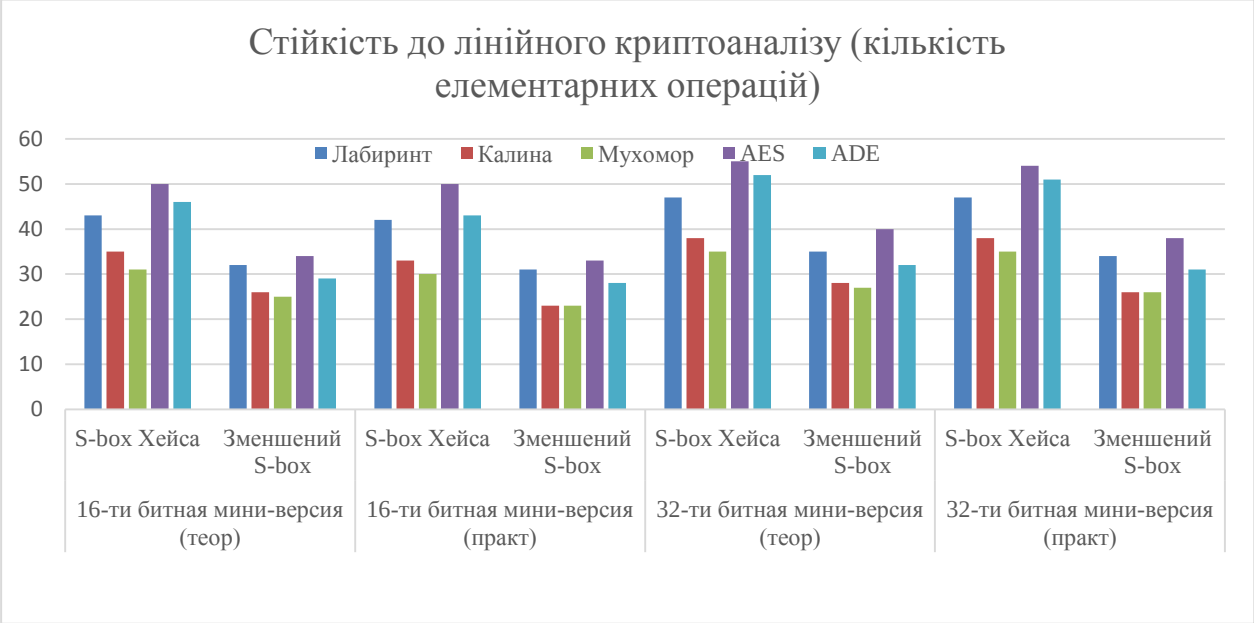


Рисунок 2.2 – Стійкість до лінійного криптоаналізу

Також можна отримати коефіцієнт зменшення для кожного алгоритму БСШ. У табл. 2.2 наведена інформація про алгоритми, коефіцієнт зменшення (*koef*), їх міні-версії та кількість операцій для відповідного алгоритму.

Таблиця 2.2 – Коефіцієнти зменшення повних версій та міні-версій шифрів

Алгоритм, коефіцієнт зменшення	Операція	Повна версія шифру	Міні-версія (16 біт)	Міні-версія (32 біти)
Лабіринт, *koef*=8	Додавання	2^{64}	2^{4}	2^{8}
	Множення	2^{64}	2^{4}	2^{8}
Калина *koef*=8	Додавання	2^{32}	2^{4}	2^{8}
	Множення	2^{64}	2^{4}	2^{8}
Мухомор *koef*=4	Додавання	2^{32}	2^{4}	2^{8}
	Множення	2^{64}	2^{4}	2^{8}
AES *koef*=4	Додавання	2^{32}	2^{4}	2^{8}
	Множення	2^{32}	2^{4}	2^{8}
ADE *koef*=4	Додавання	2^{32}	2^{4}	2^{8}
	Множення	2^{32}	2^{4}	2^{8}

Таким чином, проведений аналіз основних підходів щодо оцінки ефективності використання криптографічних алгоритмів свідчить, що важливим аспектом є мінімізація S-боксів.

2.2. Аналіз побудови S-боксів для міні-версій БСШ

Симетричні криптографічні примітиви набули широкого поширення завдяки високій продуктивності і низькій складності реалізації. S-бокси виявляються єдиним елементом, що визначає нелінійність шифрувального перетворення і рівень його стійкості до криптоаналітичних атак. Необхідна кількість раундів блокових шифрів обчислюється на основі забезпечення стійкості до відомих видів криптографічного аналізу за умови заданих властивостей вузлів нелінійної заміни [13].

Таким чином, криптографічна стійкість більшості сучасних симетричних примітивів в значній мірі залежить від властивостей обраних S-боксів. В роботах [17, 36] пропонується підхід для формування S-боксів.

В роботі [7] в якості S-боксу використовується S-бокс шифру Хейса, де використовуються міні версії S-боксів 16 і 32 бітних шифрів. Далі будуть проведені дослідження з різними версіями шифрів.

В основі всіх підходів до оцінки показників стійкості блокових симетричних шифрів до атак диференціального і лінійного криптоаналізу лежить процедура визначення максимумів середніх значень ймовірностей (максимальних середніх значень ймовірностей) повного диференціала (MADP) для всього шифру і зміщення його лінійного корпусу (MALHP).

Почнемо з Марківського процесу першого порядку, заданого рівнянням для диференціалів [18]:

$$\Delta y_r = F_r\{x\} \oplus F_r\{x^*\} = F_r^*\{\Delta x\} = F_r^*\{\Delta y_{r-1}\} \qquad (2.1)$$

В (2.1) $F_r{}^*$ – функція циклового перетворення різниць. Як випливає з рівняння (2.1), при обчисленні вихідної різниці Δy_r ключ останнього циклу k^{r+1}, як і ключі попередніх циклів наче не йдуть з рівняння. Однак це не так. Ключі поточного і попередніх циклів шифрування (випадкові компоненти) присутні в цьому перетворенні через певні значення проміжних різниць, що беруть участь у формуванні різниці шифртексту на його виході.

Проте, рішення рівняння (2.1) для фіксованого набору циклових подключей можна представити у вигляді

$$y_r \oplus y_r^* = F_r\{C_{r-1}\} \oplus F_r\{C_{r-1}^*\} = F_r^*\{\Delta C_{r-1}\}, \qquad (2.2)$$

тобто

$$\Delta y_r = F_r^*\{F_{r-1}^*\{F_{r-2}^*\{...\{F_1^*\{\Delta x_0\}\}...\}, \Delta x_0 = \Delta y_0. \qquad (2.3)$$

Можна далі ввести в розгляд повну множину значень вхідних і вихідних різниць і тоді добуток перетворень розглядати як матрицю перехідних ймовірностей, наприклад у вигляді:

$$F_r^* \cdot F_{r-1}^* \cdot F_{r-2}^* \cdot .. \cdot F_1^* = \begin{vmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,2^{n-1}} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,2^{n-1}} \\ \cdots & \cdots & : & \cdots \\ f_{2^{n-1},0} & f_{2^{n-1},1} & \cdots & f_{2^{n-1},2^{n-1}} \end{vmatrix}$$

У цій матриці кожен елемент f_{ij}, визначає ймовірність переходу i-ї різниці Δx_i на вході шифру в j-ту вихідну різницю Δy_j (в даному випадку для r циклів шифрування).

При цьому, результати розрахунків показано в табл. 2.3–2.4.

Таблиця 2.3 – Поциклові значення максимумів повних диференціалів для 16-бітних сегментів

Кількість циклів, r	Міні-Лабиринт	Міні-Калина	Міні-Мухомор	Міні-AES	Міні-ADE
1	37,5	3732,48	65536	16384	16384
2	**19,04**	382,4	5770,24	8904,25	3353,6
3	19,24	**19,36**	1802,24	1911.47	307,2
4	19,04	19,14	125,53	**19,24**	20,54
5	19,14	19,2	29,7	20,31	**19,08**
6	19,24	19,36	**18,88**	18,83	19,24
7	19,33	18,93	18,67	19,21	18,87
8	18,67	19,27	19	19,4	19,27
9	19	18,93	18	18,33	19,20
10	18	18,87	18,67	19,17	18,73

Таблиця 2.4 – Поциклові значення максимумів зсувів таблиць лінійних апроксимацій міні-шифрів зі значеннями середньоквадратичних відхилень

Кількість циклів, r	Міні-Лабіринт	Міні-Калина	Міні-Мухомор	Міні-AES	Міні-ADE

1	3178±777	9671,1±867	32768±0	16384	16384
2	980±193	3370,6±301	12839,3±1031	9284.27±657,454	9093,10±94,37
3	825,4±14	836,8±15	6400±697	818.467±26,8809	3509,8±62,37
4	825,6±23	832,2±21	1797,6±347	815±28,204	828,56±7,58
5	817,2±11	838,6±21	837,8±47	818.5±18,536	820,52±5,48
6	824±21	835,5±33	815,6±24	815.967±20,18	819,92±5,81
7	823,4±30	821,5±22	817,2±20	832.1±33,1887	818,55±5,35
8	833,6±35	827,3±18	815,8±15	823.133±23,5722	837,34±5,91
9	824,8±24	813,3±21	815,5±15	829.9±33,5741	814,95±6,21
10	819±17	834±28	810±17	827.4±25,2885	822,54±7,13

У цій дипломній роботі також будуть розглянуті результати оцінки впливу на показники сучасних шифрів (їх малих версій) максимальних значень диференціалів використовуваних S-боксів [18]. Перевірити експериментально пропоновані багатьма авторами оцінки для сучасних шифрів з бітовою довжиною вхідного блоку $n \geq 128$ обчислювально неможливо. Але це стає можливим, якщо перейти до шифрів з меншою бітовою довжиною вхідного блоку. Тому, в якості альтернативи було вибрано використовувати при оцінці показників стійкості великих шифрів зменшені їх моделі.

У табл. 2.5 наведені результати експериментів щодо визначення диференціальних показників 16-ти бітного SPN шифру (шифру з 16-ти бітовим входом) з роботи проф. Хейса [9], побудованого за ідеями Х. Фейстеля. Кожна колонка таблиці відповідає використанню S-боксів (однакових) зі своїми значеннями параметра *p*. Результати цього експерименту свідчать, що всі варіанти шифру (що відрізняються S-боксами) приходять через певну кількість циклів знову до сталого значення, характерного для випадкової підстановки.

Таблиця 2.5 – Значення повного диференціала для різних S-боксів і кількості циклів алгоритму Хейса

S-box/r	S-box AES	S-box ADE	S-box Калина	S-box Лабіринт	S-box Мухомор
1	16384,00	16384,00	24576,00	16384,00	24576,00
2	4096,00	4096,00	6144,00	4096,00	6144,00
3	2036,27	443,87	1920,00	1616,00	2802,40
4	596,00	54,60	601,20	362,87	649,33

5	190,33	25,07	148,93	88,47	292,93
6	77,47	19,00	50,00	24,93	71,47

З даної таблиці можна зробити висновок, що показники стійкості шифрів не залежать від підстановок, використаних при їх побудові, а залежать тільки від бітового розміру входу в шифр. Підстановки впливають лише на динаміку переходу до асимптотичного значення [11].

РОЗДІЛ 3 ОЦІНКА АДЕКВАТНОСТІ ВИКОРИСТАННЯ МІНІ-ВЕРСІЙ БСШ НА ОСНОВІ ВИКОРИСТАННЯ ПОВНИХ ШИФРІВ

3.1 Розробка програмного пакету для проведення можливості використання міні-версій БСШ

Для проведення досліджень був розроблений додатковий програмний продукт, який дозволяє реалізувати етапи лінійного та диференційного криптоаналізу на основі методики підр. 1.3. Він використовувався для статистичного аналізу міні-версій шифрів. Метою створення цього програмного продукту було полегшення роботи з алгоритмами.

Для проведення досліджень адекватності оцінки криптостійкості міні-версій алгоритмів-конкурсантів БСШ на стандарт України був розроблений програмний макет, який використовує міні-версії алгоритмів-конкурсантів і дозволяє проводити лінійний та диференціальний криптоаналіз, а також дослідження за допомогою пакету NIST STS 822.

Програмний макет містить 4 вкладки: Mini 16-bit, Mini 32-bit, Линейный криптоанализ и Диференціальний криптоанализ. За бажанням, можна вибрати вже готовий файл і ключ, використовуючи кнопку “Выбрать”, а потім “Открыть”.

У вкладці Mini 16-bit та Mini 32-bit можна провести такі дії:

Вибрати файл, який можна зашифрувати;

Вибрати файл в якому знаходиться ключ для шифрування;

Вибрати алгоритм за допомогою якого відбудеться шифрування;

Якщо не був вибраний файл, в якому знаходиться ключ для шифрування необхідно ввести ключ шифрування у відповідне поле;

У полі “Исходный текст” відображається інформація відкритого файлу;

У полі “Зашифрованный/Расшифрованный файл” відображається відповідна інформація.На рис. 3.1 відображено інтерфейс програмного макету.

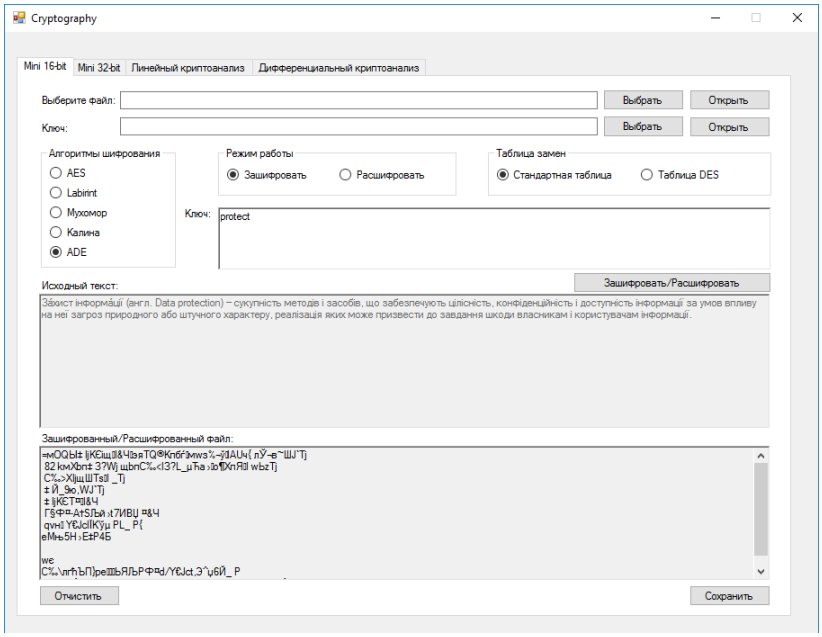
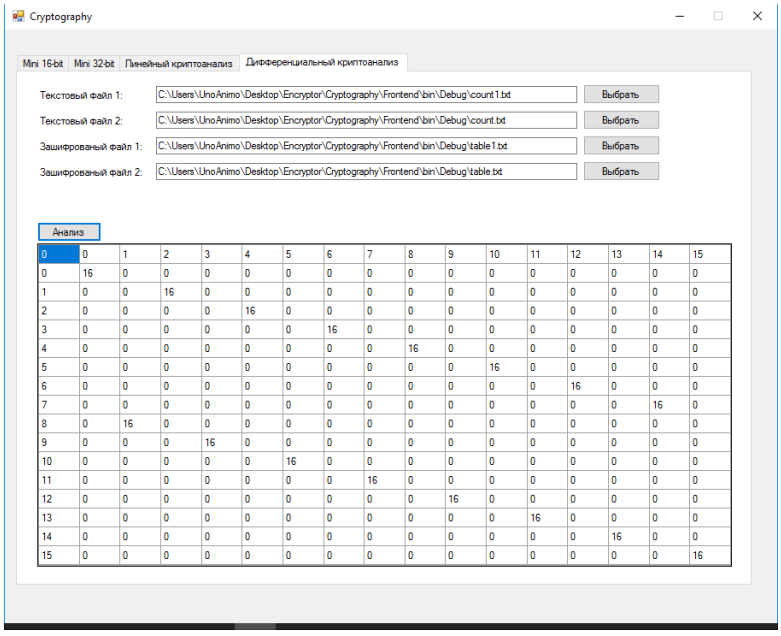


Рисунок 3.1 – Інтерфейс програмного продукту

При цьому, можна вибрати необхідний алгоритм шифрування (AES, Labirint, ADE, Калина і Мухомор), а також таблицю замін.

На рис. 3.2 відображена вкладка “Дифференциальный криптоанализ”. У цій вкладці можна вибрати текстовий та зашифрований файл для диференціального криптоаналізу. Натиснувши кнопку “Анализ” відобразиться результати криптоаналізу.



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0
8	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16

Рисунок 3.2 – Вкладка “Диференціальний криптоаналіз”

Відповідно до першого розділу можемо провести лінійний та диференціальний криптоаналіз для 16 та 32-бітних міні-версій шифрів, які використовують різні S-бокси.

У табл. 3.1 представлені оцінки стійкості до диференціального та лінійного криптоаналізу необхідних обчислювальних ресурсів (кількість елементарних операцій) для кожної 16-бітної міні-версії алгоритмів БСШ при використанні S-боксу для шифру Хейса і зменшених версій S-боксів відповідних алгоритмів, тобто зменшених на певний коефіцієнт.

Таблиця 3.1 – Обчислювальні витрати 16-бітних міні-версій шифрів

Алгоритм	Міні S-box	Стійкість до диференціального криптоаналізу (кількість елементарних операцій)	Стійкість до лінійного криптоаналізу (кількість елементарних операцій)
Лабіринт	S-box Хейса	2^{40}	2^{43}
	S-box	2^{29}	2^{32}
Калина	S-box Хейса	2^{33}	2^{35}
	S-box	2^{23}	2^{26}
Мухомор	S-box Хейса	2^{28}	2^{31}
	S-box	2^{22}	2^{25}
AES	S-box Хейса	2^{44}	2^{50}
	S-box	2^{32}	2^{34}
ADE	S-box Хейса	2^{37}	2^{46}
	S-box	2^{25}	2^{29}

У табл. 3.2 представлені оцінки стійкості до диференціального та лінійного криптоаналізу необхідних обчислювальних ресурсів (кількість елементарних операцій) для кожної 32-бітної міні-версії алгоритмів БСШ при використанні S-боксу для шифру Хейса і зменшених версій S-боксів відповідних алгоритмів, тобто зменшених на певний коефіцієнт.

Таблиця 3.2 – Обчислювальні витрати 32-бітних міні-версій шифрів

Алгоритм	S-box	Стійкість до диференціального криптоаналізу (кількість елементарних операцій)	Стійкість до лінійного криптоаналізу (кількість елементарних операцій)
Лабіринт	S-box Хейса	2^{43}	2^{47}
	Зменшений S-box	2^{35}	2^{35}
Калина	S-box Хейса	2^{37}	2^{38}

Продовження табл. 3.2

	Зменшений S-box	2^{30}	2^{28}
Мухомор	S-box Хейса	2^{33}	2^{35}
	Зменшений S-box	2^{30}	2^{27}
AES	S-box Хейса	2^{48}	2^{55}
	Зменшений S-box	2^{37}	2^{40}
ADE	S-box Хейса	2^{43}	2^{52}
	Зменшений S-box	2^{30}	2^{32}

Відповідно до експертних оцінок наведені табличні уявлення відібраних для проведення експериментів S-боксів для шифру Хейса формуються таким чином [1].

Провівши аналіз, можна зробити висновок, що 16 та 32-бітні міні-версії, які використовували S-box для шифру Хейса більш стійкі до диференціального та лінійного криптоаналізу, ніж S-бокси, зменшені на коефіцієнт.

3.2 Результати досліджень використання міні-версій для оцінки ефективності БСШ

Вихідними даними для проведення адекватності використання міні-версій є міні-версії алгоритмів, побудовані на основі конкурсантів БСШ українського конкурсу.

Лінійна таблиця апроксимацій є число рівності парності між лінійною комбінацією вхідних бітів і лінійної комбінацією вихідних бітів. При цьому, середньому значенню максимуму таблиці підстановки буде відповідати значення k *, тоді ми отримуємо найменше значення $E[\lambda(\pi, 2^k)]$, яке більше або дорівнює 1.

При цьому, потрібно визначити k^* за такою формулою:

$$\frac{(2^n-1)^2 \cdot (2^{n-1}!)^2}{2^n!} \cdot \begin{pmatrix} 2^{n-1} \\ 2^{n-2}+|k^*| \end{pmatrix}^2 = 1 \qquad (3.1)$$

Можна обчислити число елементів таблиці $LAT^* \pi$, що мають заповненням значення 2^k, як просте множення формули на загальне число елементів таблиці підстановки за виключенням першого рядка і першого стовпця:

$$E\,|\,\lambda(\pi,2k)\,|=\frac{(2^n-1)^2\cdot(2^{n-1}!)^2}{2^n!}\cdot\begin{pmatrix}2^{n-1}\\2^{n-2}+|\,k\,|\end{pmatrix}^2 \qquad (3.2)$$

Для лінійної апроксимаційної таблиці пропонується оцінювальне співвідношення, отримане на основі обробки результатів обчислювальних експериментів в [9]:

$$LP^f_{\max}\le\left(\frac{\left(\frac{3}{2}\right)^n}{2^{n-1}}\right)^2 \qquad (3.3)$$

Наведемо для цього випадку результати розрахунків для значень $n\le16$ (табл. 3.3).

Таблиця 3.3 – Порівняння розрахункових і експериментальних результатів за оцінкою максимальних значень 16 бітних міні-версій шифрів

n	*2k*	$E[\lambda(\pi,2k)]$	Експеримент
4	4 6	3,89 1,118	5,498 $\left(\frac{3}{2}\right)^8=5{,}06$
6	12 14	9,013 1,7	14,48 $\left(\frac{3}{2}\right)^6=11{,}39$
8	32 34	2,12 0,7457	34,68 $\left(\frac{3}{2}\right)^8=25{,}62$
10	74 76	1,16 0,64	78,8 $\left(\frac{3}{2}\right)^8=57{,}66$
12	162 164	1,129 0,82	116,24 $\left(\frac{3}{2}\right)^8=129{,}74$
14	350 352	1,069 0,900	$\left(\frac{3}{2}\right)^{14}=291$
16	748 750	1,027 0,93	720 $\left(\frac{3}{2}\right)^{17}=657$

З табл.3.3 видно, що знайдені значення максимумів таблиць лінійних апроксимацій випадкових підстановок добре узгоджуються з даними, отриманими експериментальним шляхом.

У правій колонці табл. 3.4 представляються результати розрахунків за спрощеною формулою, запропонованою на основі простого підбору

$$k_{\max} = 2k_D^* = n + 4 \qquad (3.4)$$

Результати свідчать, що експериментальні дані і дані розрахунків практично повторюють один одного (виявляються досить близькими).

Якщо провести експеримент і розрахувати абсолютний максимум лінійних характеристик шифрів, результати можна побачити в табл.3.4 [8]:

Таблиця 3.4 – Абсолютні значення максимумів міні-версій шифрів, представлених на український конкурс

Шифр	Mini-AES	Mini-ADE	Міні-Лабіринт	Міні-Калина	Міні-Мухомор
1	16384	16384	-	12288	12288
2	10240	12288	-	4480	9728
3	4352	4352	1444	936	8396
4	940	932	914	912	8450
5	900	896	902	900	8667
6	922	880	906	912	8696

При цьому, від 1 до 6 в табл. 3.4 показані раунди.

Проведений авторами роботи [8] аналіз говорить, що абсолютні значення максимумів практично не відрізняються від середніх значень максимумів. Але це не має відношення до шифру “Мухомор”. Це означає, що в якості максимальних значень зсувів при оцінці стійкості шифру можна використовувати поточні значення вимірювань цього показника.

Результати абсолютних значень максимумів лінійних властивостей шифрів представлені на рис.3.3:

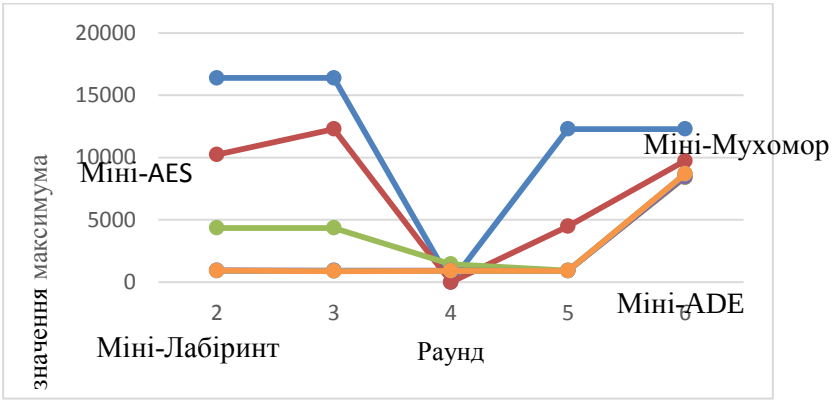


Рисунок 3.3 – Абсолютні значення максимумів лінійних властивостей міні-версій шифрів

У табл. 3.5 наведені результати розрахунків математичних очікувань максимумів міні-версій шифрів. Всі показники близькі до випадкових підстановок.

Таблиця 3.5 – Математичні очікування максимумів міні-версій шифрів

Шифр/г	AES	ADE	Лабіринт	Калина	Мухомор
1	16384	16384	-	9349,15±237,02	11602±5424,45
2	9164,8±120,75	9093,10±94,3	-	3545,8±83,93	8499,2±622,48
3	3658,2±65,8	3509,8±62,37	1069,8±38,41	830,55±83,93	7928,7±6134,3
4	827,24±7,25	828,56±7,58	826,36±6,6	820,14±6,96	7605,4±6180,14
5	821,34±6,16	820,52±5,48	820,8±5,44	823,28±4	7660,65±6234,62
6	821,68±7	819,92±5,81	822,88±7,25	825,04±6,34	7738,65±6215,21

У табл. 3.5 наведено результати абсолютних значень максимумів, де всі шифри, крім шифру Мухомор демонструють показники до випадкових підстановок (1–4).

На кожному раунді (1–6) показані значення математичних очікувань міні-версій шифрів. При цьому можна зробити висновок, що дійсно БСШ є випадковими підстановками.

Графічні результати можна побачити на рис. 3.4:

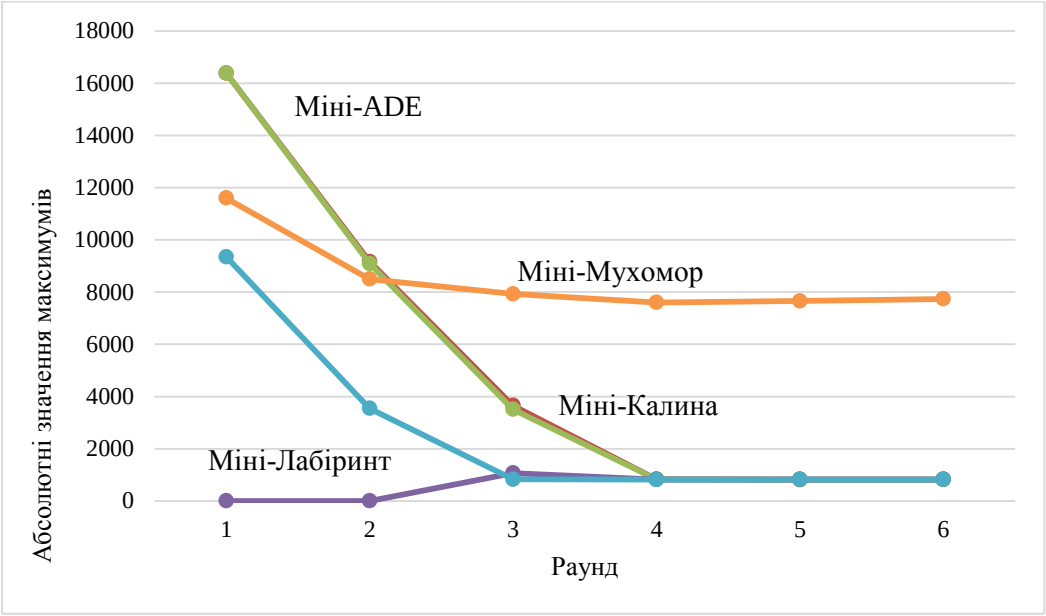


Рисунок 3.4 – Математичні очікування абсолютних значень максимумів зсувів лінійних апроксимацій

Також були виконані аналогічні розрахунки для 32-бітових версій шифрів. Звичайно, при цьому були використані інші коефіцієнти зменшення операцій. Результати наших розрахунків показані в табл. 3.6–3.7.

Таблиця 3.6 – Порівняння розрахункових і експериментальних результатів за оцінкою максимальних значень ЛАТ випадкових підстановок 32 бітних міні-версій шифрів

n	$2k$	$E[\lambda(\pi,2k)]$	Експеримент
7	10 12	2,641 0,221	10,3 ≤(n+4)
8	10 12	10,26 0,8748	11,4 ≤(n+4)
9	12 14	3,474 0,248	12,5 ≤(n+4)
10	12 14	13,8495 0,99	13,4 ≤(n+4)
11	14 16	3,952 0,247	14,5 ≤(n+4)
12	14 16	15,787 0,987	15,3 ≤(n+4)
14	16 18	15,76 0,87	17,6 ≤(n+4)
16	18 20	14,01 0,7	19,5 ≤(n+4)
32	32 34	8,155 0,239	≤(n+4)

Таблиця 3.7 – Абсолютні значення максимумів лінійних апроксимацій міні версій шифрів, представлених на український конкурс

Шифр/*r*	AES	ADE	Лабіринт	Калина	Мухомор
1	25722,88	25722,88	-	19292,16	19292,16
2	16076,8	19292,16	-	7033,6	15272,96
3	6832,64	6832,64	2267,08	1469,52	13181,72
4	1475,8	1463,24	1434,98	1431,84	13266,5
5	1413	1406,72	1416,14	1413	13607,19
6	1447,54	1381,6	1422,42	1431,84	13652,72

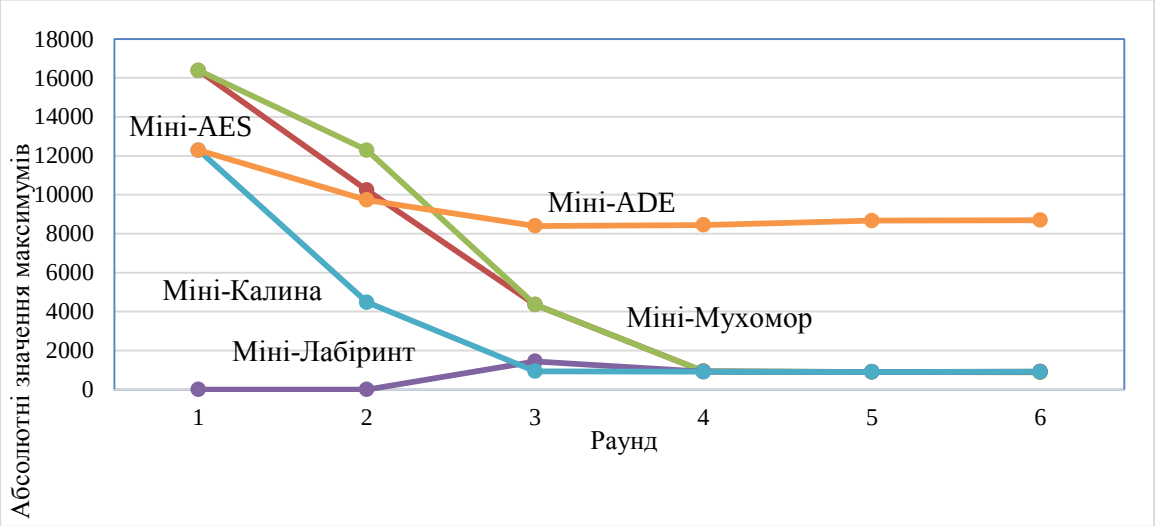


Рисунок 3.5 – Абсолютні значення максимумів лінійних властивостей міні-версій шифрів, представлених на український конкурс

Порівнюючи результати табл. 3.6–3.7 з 16-бітовими версіями, можна сказати, що результати відрізняються незначно. При цьому, шифри близькі до випадкових підстановок. Також можна помітити, що значення абсолютних максимумів в деяких випадках в 32-бітних версіях більше ніж в 16-бітних майже на 50%.

Як випливає з табл. 3.3, середнє значення максимуму таблиці лінійних апроксимацій для n = 16 лежить в діапазоні від 748 до 750. Це і є теоретична оцінка лінійної характеристики випадкової підстановки. Число раундів міні-версій шифру, при якому реалізується це асимптотичне значення, є оцінкою складності шифру для забезпечення стійкості конкретного шифру до лінійного криптоаналізу.

Таким чином, експеримент проводився для обмеженого набору ключів. Відповідні результати розглядалися як вибірка з генеральної сукупності. Результати статистичних досліджень лінійних властивостей міні-шифрів наведені в табл. 3.6–3.7.

Аналіз отриманих результатів показує, що міні-шифри, які розглядаються, виходять на асимптотичний показник середнього максимуму відхилень в такому порядку: 1) AES, 2) ADE, 3) Мухомор, 4) Калина, 5) Лабіринт. Аналогічний порядок зберігається і для абсолютного, і для середнього абсолютного максимуму. При цьому можна сказати, що міні-шифри за своїми характеристиками близькі до випадкових підстановок, однак повної відповідності між ними не досягається

3.3 Аналіз показників оцінки повних диференціалів

В роботі [43] пропонується теоретичне обґрунтування твердження, що для кожного блокового симетричного шифру (з числа відомих ітеративних БСШ) існує цілком визначене число циклів, після якого шифр набуває властивостей випадкової підстановки. Подальше нарощування числа циклів не впливає на підсумкові диференціальні і лінійні властивості шифру. Це значення є одним і тим же для всіх шифруючих перетворень з однаковим бітовим розміром входу.

Підсумкові (асимптотичні) показники стійкості (максимуми повних диференціалів таблиць XOR різниць послідовностей шифруючих перетворень також як і максимуми лінійних апроксимаційних таблиць цих же перетворень) залежать тільки від числа циклів шифрувального перетворення і розміру його бітового входу. Особливістю випадкової підстановки є те, що ми маємо справу не з фіксованим розподілом переходів різниць, а з випадковим. Таблиця XOR різниць випадкової підстановки визначається тим, що для неї є фіксованим число комірок кожного типу, що визначаються за допомогою закону розподілу $\Pr(\Lambda_f(\Delta x, \Delta y) = 2k)$ у вигляді [27]:

$$\Lambda_{m,2k} = (2^m - 1)^2 \cdot \Pr(\Lambda_f(\Delta x, \Delta y)) = \frac{(2^m - 1)^2}{2^m!} \cdot \binom{2^{m-1}}{k}^2 \cdot k! \cdot 2^k \cdot \Phi(2^{m-1} - k) \qquad (3.5)$$

Стосовно шифрувальних багатоциклових перетворень випадкових підстановок– диференціальні ймовірності DP^f повинні інтерпретуватися в позначеннях підстановлювальних перетворень для ключе-залежної функції f не як фіксовані, а як випадкові значення, що приймаються на безлічі ключів шифрування (на безлічі підстановок):

$$DP^f(\Delta x, \Delta y) = DP^f(\Delta x \to \Delta y) = \Pr(\Lambda_f(\Delta x, \Delta y) = 2k) \to DP^f(\Lambda_f(\Delta x, \Delta y) = 2k), \quad (3.6)$$

причому ці ймовірності слід вважати однаковими для всіх елементів таблиці диференціальних різниць (для всіх варіантів фіксованих поєднань вхідних і вихідних різниць).

Розрахункові співвідношення для визначення максимальних значень повних диференціалів і максимальних значень лінійних апроксимацій можуть бути отримані застосуванням законів, справедливих для випадкових підстановок, до шифрів, що розглядаються як випадкові підстановки, що і зроблено в роботах [18–19].

Стратегія, орієнтована на побудову вибірки з випадкового набору диференціальних переходів для великих шифрів не перспективна. Існує, однак, ще одна можливість формування випадкової вибірки з диференціальних переходів великого шифру. Можна аналізувати значення переходів різниць не для всієї диференціальної таблиці шифру, а тільки для її пропорційно зменшеної частини [38]. Передбачається розглядати, наприклад, диференціальну таблицю, яка утворюється при використанні шифру для шифрування не *n*-бітових блоків даних, а блоків, зменшених за розміром в 2 – 4 рази. Великий шифр використовується за типом малого для шифрування блоків даних зменшеної довжини (зашифровані блоки даних теж скорочуються до необхідного розміру), при цьому зберігаються всі перетворення і внутрішні зв’язки великого шифру.

Примітне при такому підході те, що з’являється можливість застосувати весь напрацьований апарат вивчення показників випадковості малих версій шифрів для вивчення показників випадковості великих шифрів. Пропонований

підхід дозволяє отримати два корисних результата. У першому випадку можна використовувати 16-бітні блоки відкритих і відповідних їм зашифрованих текстів. Тут ми приходимо до умов, в яких досліджувалися малі моделі шифрів [7–10]. Обчислювальних ресурсів тут вистачає для побудови всієї диференціальної таблиці (великий шифр працює, як його зменшена 16-бітна версія). Очевидно, що при даному підході можна побудувати і рядок зсувів таблиці лінійних апроксимацій. Цього мало, але вже можна перевірити збіг теоретичного і експериментального законів розподілу ймовірностей переходів в рядку (або декількох рядках) шифру і випадкової підстановки відповідного ступеня. У другому випадку можна будувати рядки диференціальної таблиці при використанні великого шифру для шифрування 32-бітних блоків даних. Для таблиць лінійних апроксимацій вдасться обчислити окремі значення осередків. Тут відкриваються перспективи для вирішення дослідницьких завдань і пошуку відповідей на питання, чи можна вважати великі шифри випадковими підстановками.

Перший експеримент був виконаний з використанням шифру Rijndael в режимі шифрування 16-бітових блоків даних. Результати цього експерименту ілюструє табл.3.8.

Таблиця 3.8 – Поциклові значення максимумів повних диференціалів при шифруванні 16-бітними блоками шифра Rijndael

Кількість циклів/ r	Значення максимуму повного диференціалу	Середньо-квадратичне відхилення
1	65536	0
2	3652,26	±630,312
3	19,066	±1,436
4	19,066	±0,997
5	18,866	±1,231
6	19,133	±0,991
7	19,266	±1,093
8	19,133	±1,431
9	19,066	±1,236
10	19,333	±1,299
11	19,4	±1,474
12	18,866	±0,991

Продовження таблиці 3.8

13	18,866	±0,991
14	18,933	±1,123

З представлених результатів видно, що шифр Rijndael вже з третього циклу шифрування приходить до сталому значенню максимуму повного диференціала, що повторює відповідне значення (рівне 19), властиве випадковий підстановці ступеня [28]. Видно, що це асимптотичне значення практично не залежить від використовуваних ключів шифрування (середньоквадратичне відхилення не перевищує 1,5). Для зменшеної до 16-бітного входу моделі шифру Rijndael асимптотичне значення (19) настає після трьох-чотирьох циклів (залежно від конструкції лінійного перетворення).

У табл. 3.9 представлені результати аналогічних експериментів, виконаних з шифром ГОСТ 28147–2009.

Таблиця 3.9 – Поциклове значення максимумів повних диференціалів під час шифрування 16-бітними блоками шифру ГОСТ28147–2009

Кількість циклів/ r	Значення максимуму повного диференціалу	Середньо-квадратичне відхилення
1	65536	0
2	65536	±12,934
3	61952	±1278,614
4	56008	±5181,74
5	31358	±857,546
6	2046,7	±637,692
7	973,4	±29,763
8	52,2	±6,808
9	19,1	±0,979
10	19,5	±2,326
11	18,7	±0,866
12	18,9	±0,994
13	19,1	±1,166
14	19,4	±0,979

Видно, що і в цьому випадку шифр стає випадковою підстановкою, але тепер для цього йому необхідно дев’ять циклів шифрування. Дев’ять циклів для великого шифру ГОСТ 28147–2009 це для нього глибина лавинного ефекту, вже давно встановлений параметр [10], який свідчить про настання моменту, з якого зміна кожного біта входу починає впливати на всі вихідні біти (при зміні будь-якого біта входу починає змінюватися в середньому половина бітів вихідного тексту). Цей ефект виявлений і в зменшеній моделі цього шифру [11], що свідчить про те, що зменшена модель шифру повторює властивості прототипу. Наступним експериментом стало обчислення закону розподілу переходів для

окремого рядка диференціальної таблиці для 32-бітного варіанта використання розглянутих шифрів.

У табл. 3.10 наведено закон розподілу ймовірностей переходів рядка диференціальної таблиці з повним числом циклів (32). У цій таблиці в лівій колонці наведені результати експериментів, а в правій – результати, отримані для випадкової підстановки ступеня розрахунковим шляхом (при 14-цикловому шифруванні). І в цьому випадку можна констатувати, що шифр Rijndael дійсно можна вважати випадковою підстановкою.

Закон розподілу переходів для рядка випадкової підстановки ступеня 2^n (у нашому випадку 2^{32}) повністю повторює закон розподілу всієї диференціальної таблиці підстановки ступеня 2^{n-1}(в даному випадку 2^{16}).

Таблиця 3.10 – Закон розподілення переходів в рядку таблиці диференційних різниць шифру Rijndael в режимі шифрування 32-бітних блоків даних

Значення переходу $2k$	Кількість переходів (експеримент)	Кількість переходів(розрахунки)
0	2605041617	$2{,}6049\times10^9$
2	1302473402	$1{,}30245\times10^9$
4	325669098	$3{,}25612\times10^8$
6	54262758	$5{,}42687\times10^7$
8	67883634	$6{,}78359\times10^6$
10	676065	678359
12	56376	56529,9
14	4089	4037,85
16	232	252,366
18	16	14,0203
20	0	0,701016

Представлені результати свідчать про те, що шифри Rijndael і ГОСТ 28147-2009 за своїми диференціальними і лінійними показниками повністю вкладаються в рамки випадкових підстановок. Експериментальні результати добре узгоджуються з теоретичними, тобто повністю підтверджується гіпотеза про те, що великі шифри, як і малі їх версії, після певного числа циклів шифрування набувають властивостей випадкових підстановок відповідного ступеня. Загальний висновок, який можна зробити з представлених матеріалів [37–41], полягає в тому, що не тільки Rijndael і ГОСТ 28147-2009, але і всі інші відомі ітеративні шифри мають властивості випадкових підстановок.

Отже, теоретичні розрахунки дозволяють визначити кількість циклів для виходу на статистичну безпеку.

3.4 Дослідження статистичних властивостей міні-версій на основі пакету NIST STS 822

Однією зі складових оцінки стійкості криптографічних примітивів є оцінка статистичної безпеки даного примітиву. Вважається, що примітив є статистично безпечним, якщо послідовність, яку він генерує, за своїми властивостями не поступається випадкової послідовності. Такі послідовності називаються “псевдо випадковими”. Для оцінки того, наскільки близько примітиву апроксимують генератори “випадкових” послідовностей, використовують статистичні тести. Запропонований NIST (американський Національним Інститутом Стандартів) пакет тестів NIST STS 828 для тестування генератора випадкових або псевдо випадкових чисел є одним з підходів реалізації завдання оцінки статистичної безпеки криптографічних примітивів. Використовуючи цей пакет тестів можна з високою часткою ймовірності зробити висновки наскільки послідовність, що генерується досліджуваним примітивом, статично безпечна.

Пакет складається з 16 статистичних тестів. Використовуючи пакет NIST STS проведемо тестування алгоритмів.

Для здійснення тестування були вибрані наступні параметри у пакеті NIST STS: довжина послідовності яка тестується $n = 10^6$ біт;

кількість послідовностей які тестуються $m = 100$. Таким чином, обсяг вибірки склав $N = 10^6 * 100 = 10^8$ біт;

рівень значущості $\alpha = 0{,}01$;

кількість тестів $q = 189$. Таким чином, статистичний портрет містить 18900 значень ймовірності P.

В ідеальному випадку при $m = 100$ та $\alpha = 0{,}01$ може бути відкинута лише одна послідовність зі 100, тобто коефіцієнт проходження кожного тесту має становити 99%. Але це занадто жорстке правило. Тому і застосовується правило на основі довірливого інтервалу для r_j. Нижня межа в цьому випадку складає значення $r_{min} = 0{,}96015$ [30]. З цих позицій проаналізуємо результати тестування алгоритмів “ADE”, “AES”, “Калина”, “Мухомор” та “Лабіринт”, наведені

статистичні портрети, утворені в результаті шифрування текстового файлу. Була побудована результуюча табл. 3.11, що наочно демонструє статистичні властивості повних версій алгоритмів та їх міні-версій алгоритмів які використовують зменшений S-box.

Проведений аналіз показав що статистичні властивості криптограм залежать від якості сформованих S-box-ів, причому результати міні-версій практично не відрізняються від результатів повних версій шифрів, не більше 20%.

Таблиця 3.11– Загальні результати тестів для зменшеного S-box та S-box для шифру Хейса

Алгоритм	Версія алгоритму		Кількість тестів, у яких тестування пройшли більше 99% послідовностей	Кількість тестів, у яких тестування пройшли більше 96% послідовностей	Кількість тестів, у яких тестування пройшли менше 96% послідовностей
Лабіринт	Повна версія		145 (76,72%)	189 (100%)	0 (0%)
	Зменшений S-box	16-біт	124 (65,61%)	189 (100%)	0 (0%)
		32-біти	133 (70,1%)	189 (100%)	0 (0%)
	S-box для шифру Хейса	16-біт	139 (73,54%)	189 (100%)	0 (0%)
		32-біти	143 (75,66%)	189 (100%)	0 (0%)
ADE	Повна версія		122 (64,55%)	188 (99,45%)	1 (0,55%)
	Зменшений S-box	16-біт	86 (45,5%)	185 (98%)	4 (2%)
		32-біти	104 (55,02%)	187 (98,9%)	2 (1,1%)
	S-box для шифру Хейса	16-біт	120 (63,49%)	187 (98,9%)	2 (1,1%)
		32-біти	122 (64,55%)	188 (99,45%)	1 (0,55%)
AES	Повна версія		133 (70,37%)	187 (98,9%)	2 (1,1%)
	Зменшений S-box	16-біт	101 (53,43%)	181 (95,76%)	8 (4,2%)
		32-біти	114 (60,04%)	187 (98,9%)	2 (1,1%)
	S-box для шифру Хейса	16-біт	132 (69,48%)	186 (98,35%)	3 (1,65%)
		32-біти	133 (70,37%)	187 (98,9%)	2 (1,1%)
Калина	Повна версія		135 (71,43%)	186 (98,4%)	3 (1,65%)
	Зменшений S-box	16-біт	107 (56,6%)	179 (94,7%)	10 (5,3%)
		32-біти	122 (64,55%)	182 (96,3%)	7 (3,7%)
	S-box для шифру Хейса	16-біт	130 (68,78%)	186 (98,35%)	3 (1,65%)
		32-біти	136 (71,95%)	187 (98,9%)	2 (1,1%)
Мухомор	Повна версія		123 (65,08%)	185 (97,9%)	4 (2,2%)
	Зменшений S-box	16-біт	87 (46,03%)	174 (92%)	15 (7,9%)
		32-біти	98 (51,85%)	183 (96,8%)	6 (3,2%)
	S-box для шифру Хейса	16-біт	123 (65,08%)	174 (98,35%)	3 (1,65%)
		32-біти	126 (66,7%)	188 (99,45%)	1 (0,55%)

При оцінці адекватності міні-версій шифрів, можна зробити висновок, що статистичні властивості міні-версій алгоритмів, які використовують S-бокс для шифру Хейса кращі, ніж міні-версії алгоритмів, які використовують зменшений S-бокс і майже однакові порівняно з повними версіями алгоритмів БСШ.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

В роботі досліджуються властивості міні-версій блочних шифрів з точки зору стійкості до різного виду атак. Світові стандарти блочних шифрів AES і DES є загально-доступними і знаходяться у вільному доступі та можуть використовуватись в будь-якій організації для шифрування даних.

Розглянемо OHSAS 18001 – британський стандарт, в якому описується розробка та впровадження систем управління охороною здоров'я і безпекою праці на підприємстві. OHSAS - це абревіатура від англійського «Occupational Health and Safety Management Systems» - що так і перекладається «Система управління гігієною та охороною праці» .

У цьому стандарті описані вимоги щодо гігієни та безпеки роботи, для того щоб, організація могла покращити свою діяльність та зменшити ризики нещасного випадку. У стандарті немає вимог щодо розробки системи управління. Вона містить у собі вимоги щодо гігієни і безпеки роботи, аніж безпека продукції та наданих послуг.

З допомогою стандарту OHSAS 18001 організація може забезпечити безпечні умови роботи, знизить кількість нещасних випадків і відповідати законодавству.

Стандарт OHSAS 18001 має багато переваг, до яких відноситься [ ]:

1. Скорочення кількості нещасних випадків і професійних захворювань.
2. Скорочення періоду простоїв і пов’язаних із ними витрат, у томи числі щодо страхування.
3. Більш ефективне управління ризиками для здоров’я та виробничої безпеки.
4. Покращення позитивного іміджу компанії в результаті прихильності до охорони праці та забезпечення виробничої безпеки.

5. Підвищення лояльності ділових партнерів і розширення кола нових клієнтів.

Стандарт OHSAS 18001 також містить у собі схему дій, заходи управління факторами, зо впливають на виникнення небезпечних ситуацій.

Порядок управління документацією з охорони праці містить кілька розділів [ ]:

1. Розділ «Область застосування».
2. Розділ «Термін та визначення».
3. Розділ «Позначення та скорочення».
4. Розділ «Вимоги».

У розділ «Область застосування» мають входити вимоги щодо управління документацією та дія має розповсюджуватися на всі підрозділи. У документації мають проходити зміни якщо є нові затвердженні, скасовані чи введені в дію документи, надходження нового документа щодо управління охороною праці в організації, або ж надходження відповідних вказівок або розпоряджень від керівництва.

У розділі «Термін та визначення» мають бути описані найчастіше вживані терміни з їхніми визначеннями.

Розділ «Позначення та скорочення» містить у собі абревіатури з їхніми визначеннями та скорочення.

У розділі «Вимоги» описаний порядок управління документацією з охорони праці. Порядок має свої положення та визначає реалізацію:

1. На підприємстві встановлюється певна ієрархія документації: Положення про систему управління охороною праці.
2. Процедурні документи з описом операцій, які оформлюються у визначеній формі.
3. Інструктивні документи.
4. Записи та інша супутня документація (колективний договір, програма заходів з управління, плани роботи підприємства з питань, суміжних із

забезпеченням охорони праці, службові записки, переліки, реєстри, запити та відповіді на них тощо).

У записи та іншої документації з охорони праці можна включати акти про нещасні випадки та професійні захворювання, додаткові матеріали щодо інциденту, протоколи нарад, результати перевірки знань з охорони праці, картки видачі індивідуального захисту, звіти, результати атестації робочих місць, оцінювання ризиків, аудит та матеріали перевірок керівництвом.

Ведення записів дозволяє аналізувати результативність, ефективність впровадження системи управління та вибрати заходи щодо вдосконалення, виявляти причини та проаналізувати невідповідності, створити свій метод одержання достовірної інформації.

Записи мають вестися як і на паперових, так і на електричних носіях для подальшого аналізу та при необхідності бути підтвердженими у юридичних осіб. Основним нюансом є те, що записи не можуть бути зміненими.

Якщо організація буде виконувати вимоги та відповідати створеному OHSAS 18001, то це автоматично знизить ризик бути оштрафованими або потрапити під правову відповідальність і судові розгляди, якщо виникнуть нещасні випадки. Впровадження OHSAS 18001 – це довгострокова стратегія щодо безпеки працівників.

4.2. Безпека в надзвичайних ситуаціях

Великі аварії на хімічно небезпечних об’єктах є одними з найбільш небезпечних технологічних катастроф, які можуть призвести до масового отруєння і загибелі людей і тварин, значного економічного збитку і важких екологічних наслідків. Причини аварій, в більшості випадків, пов’язані з порушеннями встановлених норм і правил при проектуванні, будівництві і реконструкції хімічно небезпечних об’єктів, порушенням технології виробництва, правил експлуатації обладнання, машин і механізмів, апаратів, низької трудової і технологічної дисципліни виробничого процесу.

Хлор за обсягом виробництва і галузі застосування є одним з найважливіших продуктів хімічної промисловості. Широке використання і великі обсяги виробництва хлору визначають високу потенційну небезпеку виникнення надзвичайних ситуацій, обумовлених його аварійними викидами в навколишнє середовище. Ці обставини поглиблюються фізико-хімічними та токсикологічними властивостями хлору, що є сильнодіючою отруйною речовиною задушливого характеру. Токсикологічні та фізико-хімічні властивості хлору є основними вражаючими чинниками при його аварійних викидах.

Комплекс заходів щодо зберігання і використання хлору включає [ ]:

- використання безпечних технологій;
- здійснення організаційних, технічних та інших заходів, що забезпечують високу експлуатаційну надійність об’єктів, а також обмеження розповсюдження хлору за межі санітарно-захисної зони при аваріях і руйнуваннях;
- раціональне розміщення хлору з урахуванням можливих наслідків аварій;
- підготовка і проведення спеціальних заходів щодо захисту населення, що дозволяють знизити масштаби шкідливого впливу.

Велике значення в профілактиці аварій з викидом хлору має оснащеність цих підприємств швидкодіючими технічними засобами захисту, в тому числі автоматичним відсічними пристроями, системами вибухопопередження і локалізації розвитку аварій, а так само відповідною підготовкою персоналу.

Ефективним способом зменшення наслідків аварій є зниження запасів хлору до мінімальної, необхідної за технологією, кількості. Особливо це важливо на етапах вантажно-розвантажувальних робіт, в сховищах хлору і готової продукції. Доцільно проводити роботи, спрямовані на створення таких умов зберігання хлору, які дозволяють виключити можливість його залпових викидів у великих обсягах.

Стабільність експлуатації об’єктів з хлором і його похідними повинна забезпечуватися високою надійністю електропостачання, та використанням

систем безаварійної зупинки при припиненні подачі електроенергії. Для підвищення міцності обладнання може проводитися обвалювання, заглиблення в ґрунт або розміщення під землею. Навколо великих сховищ доцільно споруджувати захисні оболонки.

При гострому отруєнні хлором виникає токсичний ларингіт, бронхіт, в більш важких випадках – набряк легень, пневмонія. Вдихання концентрованих парів хлору викликає хімічний опік верхніх дихальних шляхів і може привести до рефлекторної зупинки дихання [ ].

У клінічній картині, що розвивається при отруєнні хлором, виділяють період роздратування (рефлекторний період), обумовлений дратівливою дією хлору на слизову дихальних шляхів, очі. При цьому виникає відчуття печіння і дряпання в дихальних шляхах, відчуття утруднення дихання, різь в очах, слинотеча.

Одним з грізних проявів ураження хлором є розвиток токсичного набряку легень. Причиною його є підвищення проникності капілярної і альвеолярної стінки. Токсичний набряк легень виникає як в результаті безпосереднього впливу хлору на легеневу тканину, так і в результаті загальних розладів в організмі.

Перша допомога ураженому хлором полягає в наступному:

- одягання на потерпілого промислового протигаза типу В або громадянського ДП-5, ГП-7;
- винесення потерпілого на незаражену територію і зняття протигаза;
- звільнення від тісного одягу;
- при відсутності дихання – штучне дихання, переважно методом “рот в рот”;
- вдихання, для пом’якшення подразнення, аерозолю 0,5% розчину соди, а також кисню;
- промивання шкіри і слизових оболонок 2% содовим розчином;
- рясне питво (тепла вода з содою, чай, кава);

- максимальне обмеження самостійного пересування потерпілого, подальше транспортування тільки в лежачому положенні;
- у холодну пору – відігрівання і забезпечення повного спокою;
- накласти асептичні пов’язки на рани і іммобілізувати пошкоджені кінцівки;
- евакуювати уражених у медичні пункти для надання першої лікарської допомоги та подальшого лікування.

ВИСНОВКИ

Проведений аналіз реалізації міні-версій і їх використання при оцінці криптостійкості повних шифрів показав, що основною складністю при використанні міні-версій є формування зменшеної версії S-box-ів, яка забезпечує основні показники збалансованості, нелінійності і автокореляції як і в повному шифрі. Залишається невирішеним питання вибору коефіцієнта, що вимагає додаткових досліджень, який визначає зменшення операцій шифрування і потребує експертного підходу.

Проаналізовано вимоги щодо побудови БСШ та основні методи оцінки криптостійкості БСШ. У роботі представлені результати досліджень повних диференціалів, що дозволяють визначити теоретично кількість циклів криптоперетворень до виході шифру на статистичну безпеку.

Розроблений програмний продукт підтверджує результати теоретичних досліджень та розрахунків. На основі отриманих результатів проведені дослідження статистичних властивостей криптограми за допомогою пакету NIST STS 822. Отримані результати підтверджують основні висновки по використанню міні-версій БСШ для оцінки криптостійкості повних версій БСШ.

Проведені дослідження дозволяють визначити, що основними вимогами до використання міні-версій БСШ щодо оцінки криптостійкості повних шифрів є експертна оцінка коефіцієнту зменшення логічних операцій в повних шифрах, адекватність побудови S-box-ів для міні-версій БСШ щодо забезпечення основних показників збалансованості, нелінійності, автокореляції.

СПИСОК ЛІТЕРАТУРИ

1. Bernstein D.J. Cache-timing attacks on AES [Електроний ресурс] / D.J. Bernstein. – Режим доступу: http://cr.yp.to
2. Howard M.S. A Tutorial on Linear and Differential Cryptanalysis / M.S. Howard. – Electrical and Computer Engineering Faculty of Engineering and Applied Science Memorial University of Newfoundland St. John's, NF, Canada
3. D'Agapeyeff A. S. Codes and Ciphers. A History of Cryptography [Electonic resource]/ A.S. D'Agapeyeff. – Access mode: https://www.amazon.com/Codes-Ciphers-Cryptography-Alexander DAgapeyeff/dp/1406798584
4. Digital Security [Електроний ресурс] – Режим доступу: http://www.dsec.ru
5. Junod P. K. Advanced Linear Cryptanalysis of Block and Stream Ciphers (Cryptology and Information Security) [Text] / P. K. Junod, A. S. Canteaut [Electonic resource]. – Access mode: https://www.amazon.com/Advanced-Cryptanalysis-Cryptology-Information-Security/dp/1607508435
6. Klimov A.S. Analysis of Neural Cryptography [Text] / A.S. Klimov, A.P. Mityaguine, and A.V. Shamir // Computer Science department, The Weizmann Institute, Rehovot 76100 Israel.
7. Krzysztof C.H. On Differential and Linear Approximation of S-box Functions / Biometrics, Computer Security Systems and Artificial Intelligence Applications. / Khalid Saeed, Jerzy Pejas and Romuald Mosdorf. // Poland, Springer – 2006. – P. 111-120.
8. Lagun, A. S. Cryptographic strength of a new symmetric block cipher based on Feistel network [Text] / A. S. Lagun // Technical transactions automatic control. – 2013. – pp. 68-80.
9. Мао, В.О. Современная криптография: Теория и практика / В.О. Мао. — М.: Вильямс, 2005. — 768 с.
10. National Institute of Standards, Advanced Encryption Standard (AES) [Электронный ресурс]. – Режим доступа: www.nist.gov/aes/
11. Nechvatal J. R. Report on the Developmentof the Advanced Encryption Standard (AES) [Text] / J. R. Nechvatal, E. M. Barker, L.O. Bassham, W. A. Burr, M. D. Dworkin, J. L. Foti, E. A. Roback // Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Technology Administration, U.S. Department of Commerce. – p.116.

12. Oliynykov R. V. Results of Ukrainian national public cryptographic competition / R. V. Oliynykov, I. O. Gorbenko, V. A. Dolgov, V. B. Ruzhentsev // Tatra Mt. Math. Publ. 47 . – 2010 . – pp. 99–113.

13. Rukhin A.S., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST STS Special Publication 800-22 // A.S. Rukhin, J.V. Soto., 09.2000.

14. Sahu H. K. Cryptanalytic attacks on international data encryption algorithm block cipher / H. K. Sahu, V.A. Jadhav , S. S. Sonavane, R.K. Sharma // Defence Science Journal, Vol. 66, No. 6. – 2016. – pp. 582-589.

15. Столлингс В. В. Криптография и защита сетей: принципы и практика [Текст] / В.В. Столлингс. – М.: Издательский дом ”Вильямс”, 2001.

16. Фергюсон, Н.А. Практическая криптография. Practical Cryptography: Designing and Implementing Secure Cryptographic Systems: учеб./ Н.А. Фергюсон, Б.Ю. Шнайер. – М. : Диалектика, 2004. – 432 с. – 3000 экз. –ISBN 5-8459-0733-0, ISBN 0-4712-2357-3.

17. Weinmann, R. P. Algebraic Methods in Block Cipher Cryptanalysis: dis. Dr. rer. nat. / R. P. Weinmann. – 2009. – 113 p.

18. William Stalings Cryptography and Network Security: Principles and Practice (7th Edition) [Electonic resource]. – Access mode: https://www.pearsonhighered.com/program/Stallings-Cryptography-and-Network-Security-Principles-and-Practice-7th-Edition/PGM334401.htm

19. Алгоритмы шифрования [Электронный ресурс]. – Режим доступа: http://www.ixbt.com/soft/alg-encryption-aes-2.shtml

20. Алексийчук А. Н. Оценки практической стойкости блочного шифра “Калина” относительно методов разностного, линейного криптоанализа и относительно алгебраических атак, основанных на гомоморфизмах / А. Н. Алексийчук, Л. В. Ковальчук, Е. В. Скрыпник, А. С. Шевцов // Прикладная радиоэлектроника. – 2008. – Т. 7, № 3. – С. 203-209.

21. Бабенко Л.К. Особенности дифференциального криптоанализа алгоритма AES [Электронный ресурс] / Л.К.Бабенко, Е.А.Ищукова .– Режим доступа: http://www.contrterror.tsure.ru/site/magazine8/07-22-Babenko.htm

22. Белецкий А.Я. Семейство симметричных блочних RSB криптографических алгоритмов с динамически управляемыми параметрами шифрования / А.Я. Белецкий, А.А. Белецкий, А.А. Кузнецов // Електроніка та системи управління. – 2007. – № 1 (11). – С. 5-16.

23. Бессмертный И. А. Методические указания по подготовке магистерской диссертации / И. А. Бессмертный. – СПб : ФГОУВПО "СПбГУИТМО", 2011. – 102 с.

24. Головашич С.А. Спецификация алгоритма блочного симметричного шифрования “Лабиринт”/ С.А. Головашич // Прикладная радиоэлектроника. – 2007. – Т. 6, № 2. – С. 230-240.

25. Горбенко І.Д. Перспективний блоковий симетричний шифр “Мухомор”. Основні положення та специфікація / І.Д. Горбенко, В.І. Долгов, Р.В. Олійников та ін. // Прикладная радиоэлектроника. – 2007. – Т. 6, № 2. – С. 147-157.

26. Горбенко, И. Д. Новая идеология оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа / И. Д. Горбенко, В. И. Долгов, И. В. Лисицкая, Р. В. Олейников // Прикладная радиоэлектроника. – 2010. – Т. 9, № 3. – С. 212-320.

27. Горбенко, И.Д. Новая идеология оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа / И. Д. Горбенко, В.И. Долгов, И. В. Лисицкая, Р.В. Олейников // Прикладная радиоэлектроника, том 9, № 3. – 2010. – с. 312-320.

28. Дифференциальный криптоанализ [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%84%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7

29. Долгов, В.И. Линейные свойства блочных симметричных шифров, представленных на украинский конкурс / В.И. Долгов, А.А. Кузнецов, С.А. Исаев // Электрон. моделирование. – 2011. – с. 81-99.

30. Долгов, В. И. Свойства таблиц линейных аппроксимаций случайных подстановок / В. И. Долгов, И. В. Лисицкая, О. И. Олешко // Прикладная радиоэлектроника. – 2010. – Т. 9, № 3. – С. 334–340.

31. Долгов, В. И. Криптографические свойства уменьшенной версии шифра “Калина” / В. И. Долгов, Р.В. ОЛЕЙНИКОВ, А.Ю. Большаков // Прикладная радиоэлектроника, Том 9, № 3 . – 2010. – с. 349-354.

32. Кузнецов О.О Захист інформації в інформаційних системах.методи традиційної криптографії навчальний посібник // Кузнецов О.О, Євсеєв С.П.,Король О.Г., Харків. Вид. ХНЕУ, 2010.

33. Інформаційна безпека [Електроний ресурс]. – Режим доступу: http://www.ua7.org/mexanizmi-informacijnoї-bezpeki

34. Исаев, С.А. Электронное моделирование / С.А. Исаев. – 2011. – Т. 33, № 6. – С. 81–99.

35. Линейный криптоанализ [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D1%8B%D0%B9_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7

36. Лисицкая И.В. Методология оценки стойкости блочных симметричных шифров / И. В. Лисицкая // Автоматизированные системы управления и приборы автоматики. – 2011. – № 163. – С. 123-133.

37. Лисицкая И.В. Анализ дифференциальных и линейных свойств шифров rijndael, serpent, threefish при 16-битных входах и выходах / И.В. Лисицкая, Т.А. Гриненко, С.Ю. Бессонов // Восточно-Европейский журнал передовых технологий ISSN 1729-3774. – 2015, с. 50-54.

38. Лисицкая И.В. О новой методике оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа / И.В. Лисицкая // Системи обробки інформації, випуск 4 (94). – 2011. – с. 167-173

39. Лисицкая И.В. Криптографические свойства уменьшенной версии шифра Мухомор. / И.В. Лисицкая, О.И. Олешко, С.Н. Руденко, Е.В. Дроботько, А. В. Григорьев // Збірник наукових праць. – Вип. Київ, 2010. с. 31-42.

40. Лисицкая И.В. Большие шифры и случайные подстановки / И.В. Лисицкая, А.А. Настенко // ISSN 0485-8972 Радиотехника, 2011.

41. Лисицкий К.Е. Максимальные значения полных дифференциалов и линейных корпусов блочных симметричных шифров / К.Е. Лисицкий // Технологический аудит Technology audiT and producTion reserves — № 1/1(15) – 2014. – с. 47-52

42. Лисицкая И. В. Методология оценки стойкости блочных симметричных шифров [Текст] / И. В. Лисицкая // Автоматизированные системы управления и приборы автоматики. – 2011. – № 163. – с. 123–133.

43. Лисицкая И. В. Сравнение по эффективности суперблоков некоторых современных шифров [Текст] / И. В. Лисицкая // Радіоелектроніка. Інформатика. Управління. — Запоріжжя, 2012. – № 1(26). – с. 37-43.

44. Лисицкая И.В. Методология оценки стойкости блочных симметричных криптопреобразований на основе уменьшенных моделей [Текст] : дисс.докт. тех. наук: 05.13.05 / И. В. Лисицкая. – 2012. – 293 с.

45. Лисицкая И.В. 32-х битная мини-версия блочного симметричного алгоритма криптографического преобразования информации Мухомор. Оценка макимального значения полного дифференциала шифра [Текст] / И.В. Лисицкая, И.А. Ставицкий // Автоматизированные системы управления и приборы автоматики. – 2011. – № 7(102). Выпуск18/1. – с. 11-21.

46. Методичні вказівки до виконання магістерської атестаційної роботи за спеціальностями 8.05010301 "Програмне забезпечення систем", 8.05010302 "Інженерія програмного забезпечення" для студентів усіх форм навчання / укл. З. В. Дудар, В. І. Каук, І. А. Ревенчук та ін. – Х. : ХНУРЕ, 2011. – 51 с.

47. Олейников, Р. В. Дифференциальные свойства подстановок / Р. В. Олейников, О. И. Олешко, К. Е. Лисицкий, А. Д. Тевяшев // Прикладная радиоэлектроника. – 2010. – Т. 9, № 3. – с. 326–333.

48. Олейников Р.В. Исследование дифференциальных свойств подстановок / Р.В. Олейников, И.В. Лисицкая, А.В. Широков, К.Е. Лисицкий // Компьютерные науки и технологии: сб. научн. тр. первой межд. НТК – Ч. I. – Б., 2009. – С. 59-63.

49. Семенов, Ю.А. Алгоритм шифрования AES / Ю.А. Семенов [Электронный ресурс]. – Режим доступа: http://book.itep.ru/6/aes.htm

50. Скиба Ю.Д. Аналіз конкурсів блочного симетричного шифрування і алгоритмів-переможців // Актуальні проблеми науки та освіти молоді: теорія, практика, сучасні рішення-2010: (Матеріали міжнародної науковопрактичної конференції молодих вчених, аспірантів та студентів) [Електронний ресурс] : ред. В.С. Пономаренко, О.І. Пушкар. – Х.: вид. ХНЕУ, 2010. – 1 електрон. опт. диск (CD-ROM) : кольор. ; 12 см. – Систем. вимоги: Pentium; 32 Mb RAM ; CD-ROM Windows 98/2000/NT/XP; Adobe Acrobat Reader.

Сорока, Л.С. Исследование дифференциальных свойств блочно-симметричных шифров / Л.С.Сорока, А.А.Кузнецов, И.В.Московченко, С.А. Исаев // Системы обработки информации. – 2010. – С. 286-295.

ДОДАТКИ

Додаток А

Статистичні портрети блоково-симетричних шифрів (міні-версії)

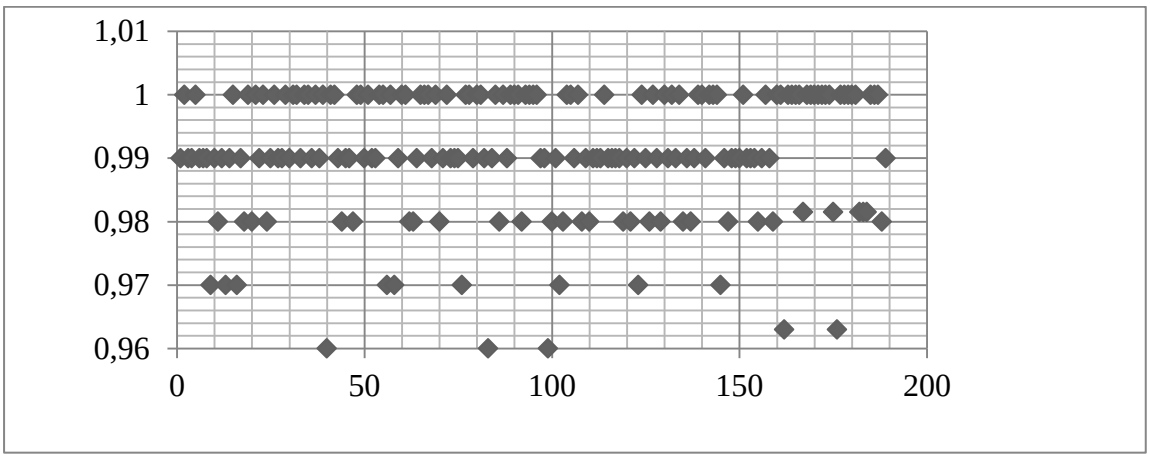


Рисунок А.1 – Статистичний портрет алгоритму “Лабіринт”

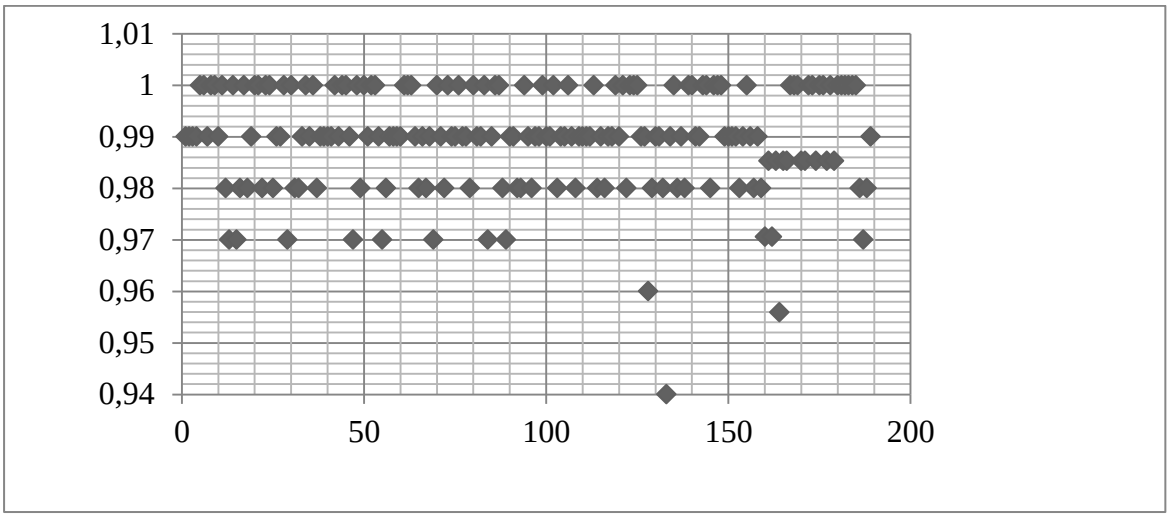


Рисунок А.2 – Статистичний портрет алгоритму “AES”

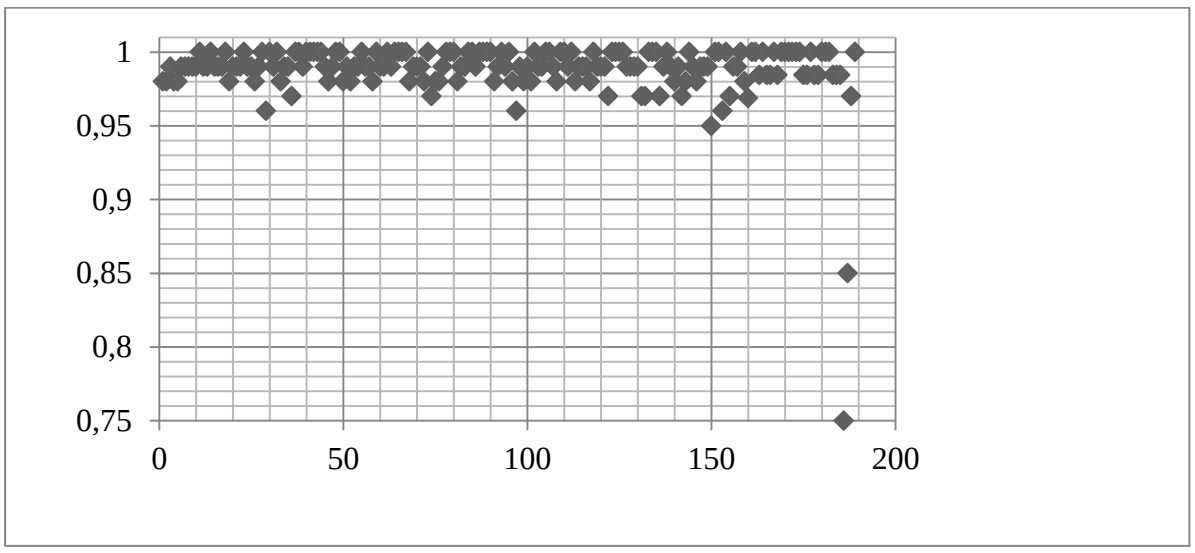


Рисунок А.3 – Статистичний портрет алгоритму “Калина”

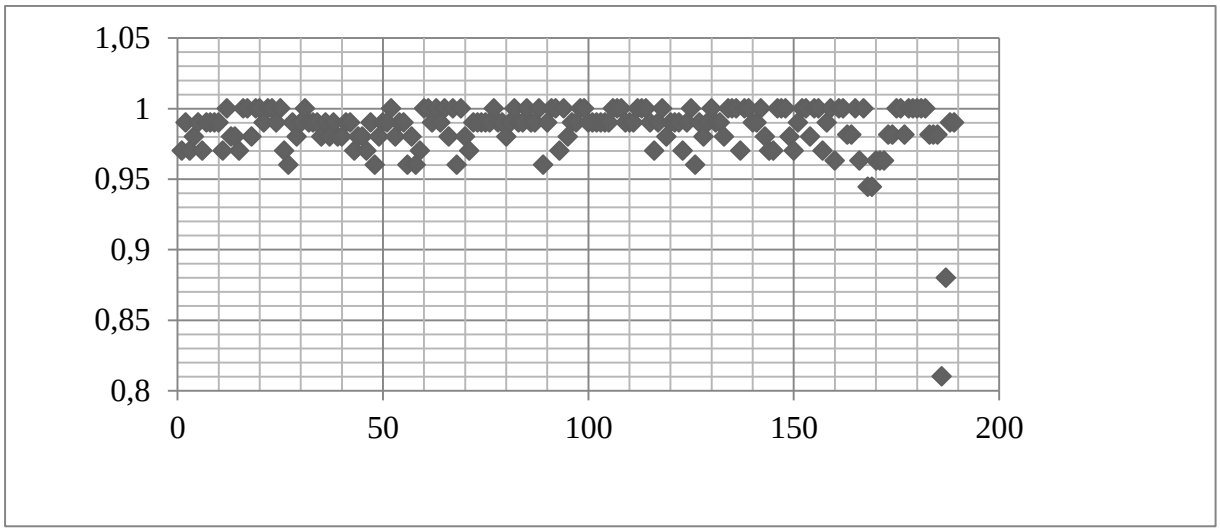


Рисунок А.4 – Статистичний портрет алгоритму ”Мухомор”

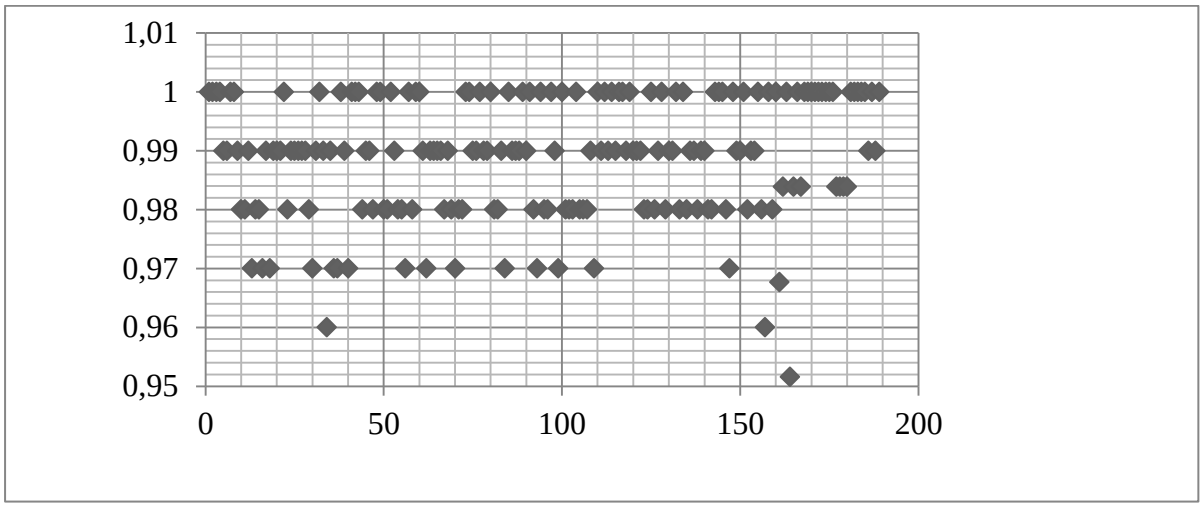


Рисунок А.5 – Статистичний портрет алгоритму “ADE”

Додаток Б

Значення максимуму S-бокси

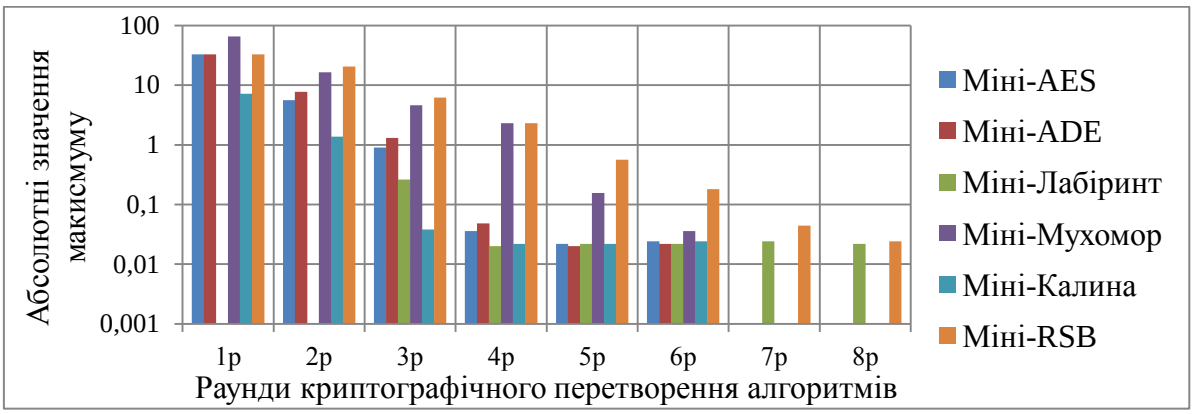


Рисунок Б.1 – Абсолютні значення максимуму, S-бокси з NL = 2

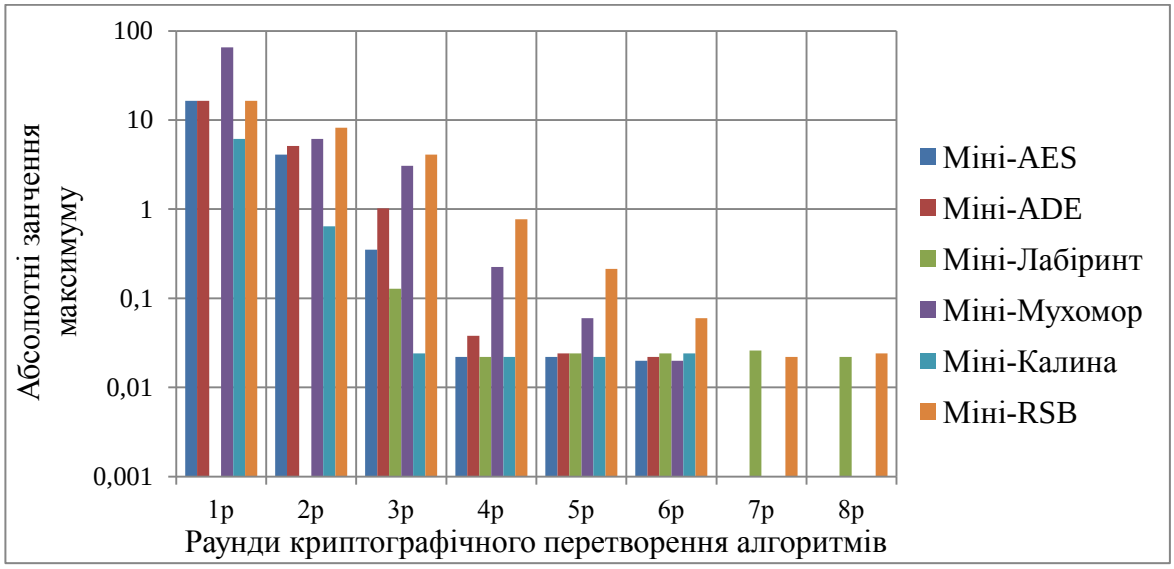


Рисунок Б.2 – Абсолютні значення максимуму, S-бокси з NL = 4

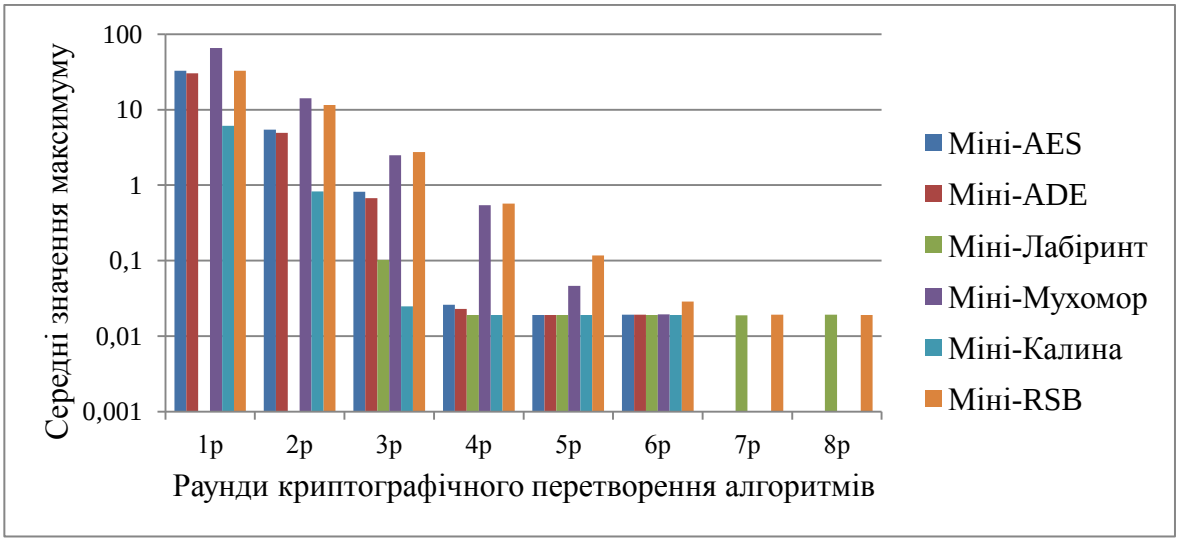


Рисунок Б.3 – Середні значення максимуму, S-бокси з NL = 2

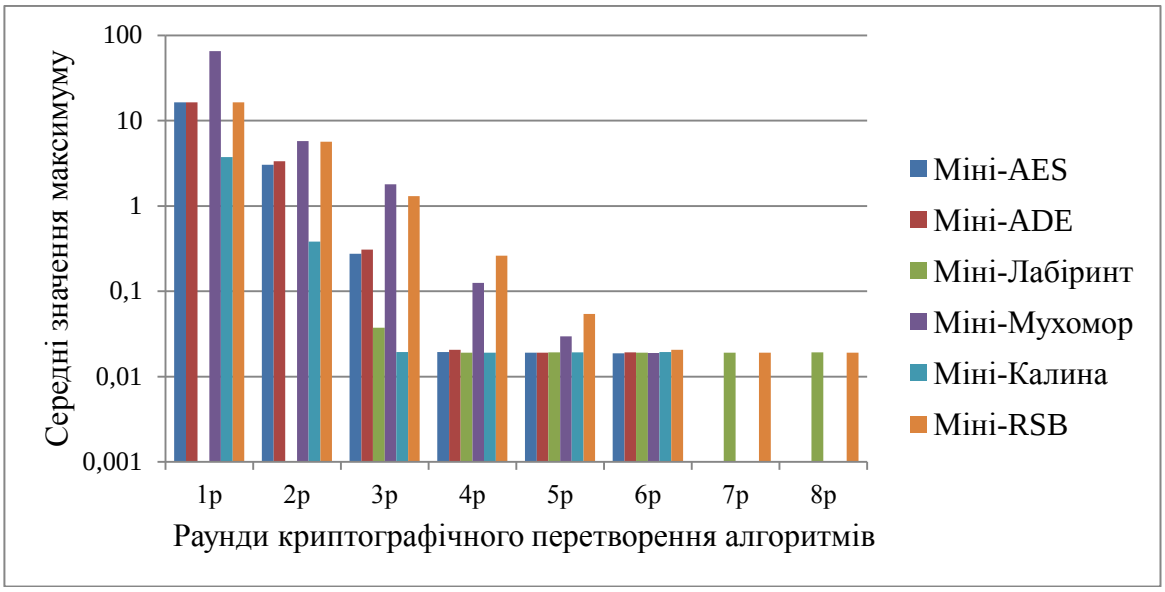


Рисунок Б.4 – Середні значення максимуму, S-бокси з NL = 4

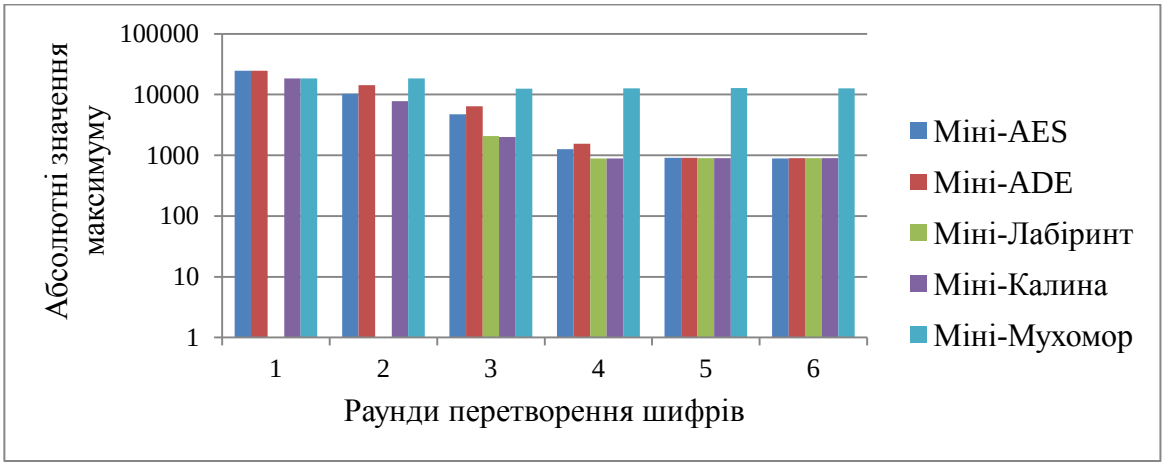


Рисунок Б.5 – Абсолютні значення максимуму, S-бокси з NL = 2

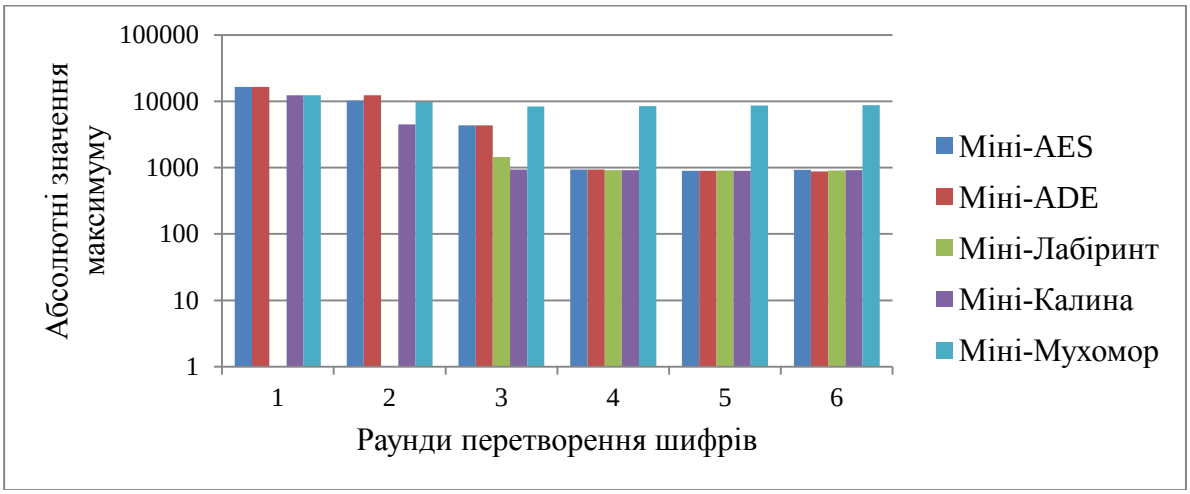


Рисунок Б.6 – Абсолютні значення максимуму, S-бокси з NL = 4

Додаток В -Лістинг програми

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 14
VisualStudioVersion = 14.0.25420.1
MinimumVisualStudioVersion = 10.0.40219.1
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "Form", "Frontend\Form.csproj", "{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}"
EndProject
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "Editor", "Editor\Editor.csproj", "{2BDBB3A9-A052-4E5F-A982-18343776F218}"
EndProject
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "ScramblerLogic", "ScramblerLogic\ScramblerLogic.csproj", "{CE3ADDA6-F4E2-40D5-B940-4AD0CACC48A1}"
EndProject
Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "Cryptanalysis", "Cryptanalysis\Cryptanalysis.csproj", "{C1D45B23-45D9-4980-B891-C6CBD4F44426}"
EndProject
Global
GlobalSection(SolutionConfigurationPlatforms) = preSolution
Debug|Any CPU = Debug|Any CPU
Debug|x64 = Debug|x64
Debug|x86 = Debug|x86
Release|Any CPU = Release|Any CPU
Release|x64 = Release|x64
Release|x86 = Release|x86
EndGlobalSection
GlobalSection(ProjectConfigurationPlatforms) = postSolution
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|Any CPU.Build.0 = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|x64.ActiveCfg = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|x64.Build.0 = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|x86.ActiveCfg = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Debug|x86.Build.0 = Debug|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|Any CPU.ActiveCfg = Release|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|Any CPU.Build.0 = Release|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|x64.ActiveCfg = Release|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|x64.Build.0 = Release|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|x86.ActiveCfg = Release|Any CPU
{4AB8E586-DEA0-4C63-AACD-F37EDB8A1368}.Release|x86.Build.0 = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-18343776F218}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
{2BDBB3A9-A052-4E5F-A982-18343776F218}.Debug|Any CPU.Build.0 = Debug|Any CPU
{2BDBB3A9-A052-4E5F-A982-18343776F218}.Debug|x64.ActiveCfg = Debug|Any CPU
{2BDBB3A9-A052-4E5F-A982-18343776F218}.Debug|x64.Build.0 = Debug|Any CPU
{2BDBB3A9-A052-4E5F-A982-18343776F218}.Debug|x86.ActiveCfg = Debug|Any CPU
```

{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Debug|x86.Build.0 = Debug|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|Any CPU.ActiveCfg = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|Any CPU.Build.0 = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|x64.ActiveCfg = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|x64.Build.0 = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|x86.ActiveCfg = Release|Any CPU
{2BDBB3A9-A052-4E5F-A982-
18343776F218}.Release|x86.Build.0 = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|Any CPU.Build.0 = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|x64.ActiveCfg = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|x64.Build.0 = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|x86.ActiveCfg = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Debug|x86.Build.0 = Debug|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|Any CPU.ActiveCfg = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|Any CPU.Build.0 = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|x64.ActiveCfg = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|x64.Build.0 = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|x86.ActiveCfg = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
4AD0CACC48A1}.Release|x86.Build.0 = Release|Any CPU
{CE3ADDA6-F4E2-40D5-B940-
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|Any CPU.ActiveCfg = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|Any CPU.Build.0 = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|x64.ActiveCfg = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|x64.Build.0 = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|x86.ActiveCfg = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Debug|x86.Build.0 = Debug|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Release|Any CPU.ActiveCfg = Release|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Release|Any CPU.Build.0 = Release|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Release|x64.ActiveCfg = Release|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Release|x64.Build.0 = Release|Any CPU
{C1D45B23-45D9-4980-B891-
C6CBD4F44426}.Release|x86.ActiveCfg = Release|Any CPU
{C1D45B23-45D9-4980-B891-

```
C6CBD4F44426}.Release|x86.Build.0 = Release|Any CPU
                                                EndGlobalSection
                                        GlobalSection(SolutionProperties) = preSolution
                                                                                HideSolutionNode = FALSE
                                                EndGlobalSection
EndGlobal
```

Реалізація алгоритмів шифрування:

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 14
VisualStudioVersion = 14.0.25420.1
MinimumVisualStudioVersion = 10.0.40219.1
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "ГОСТ_28147-89", "ГОСТ_28147-
89\ГОСТ_28147-89.vcxproj", "{A3B39452-3ED5-4CF6-A893-52F36999C458}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Kalina", "Kalina\Kalina.vcxproj",
"{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Labirint", "Labirint\Labirint.vcxproj",
"{F089E784-C194-4208-A2C6-58879F188406}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Mukhomor", "Mukhomor\ADE.vcxproj",
"{4821031C-D418-462B-8B51-78BFD0CD2163}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "ADE_Project",
"ADE_Project\ADE_Project.vcxproj", "{E58755B0-8E17-4B29-818D-99077002FA98}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "BlockReader",
"BlockReader\BlockReader.vcxproj", "{EC883BCB-9D97-4709-B67D-62DC7405C728}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Mukhomor32",
"Mukhomor32\Mukhomor32.vcxproj", "{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "AES32", "AES32\AES32.vcxproj",
"{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Kalina32", "Kalina32\Kalina32.vcxproj",
"{0F7AD900-FCCE-40CF-942D-A6EB769432E7}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "ADE32", "ADE32\ADE32.vcxproj",
"{BD820F82-1905-4463-9A61-28619EB34D02}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Labirint32", "Labirint32\Labirint32.vcxproj",
"{92161C45-0F41-4688-A4EE-3C7084883355}"
EndProject
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "AES", "AES\Aardvark.vcxproj",
"{D65CA7DC-7FAF-4C43-BB82-62F4EB0ED55F}"
EndProject
Global
                        GlobalSection(SolutionConfigurationPlatforms) = preSolution
                                                        Debug|Any CPU = Debug|Any CPU
                                                Debug|Mixed Platforms = Debug|Mixed Platforms
                                                        Debug|Win32 = Debug|Win32
                                                Release|Any CPU = Release|Any CPU
                                        Release|Mixed Platforms = Release|Mixed Platforms
                                                        Release|Win32 = Release|Win32
                                                EndGlobalSection
```

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Any CPU.ActiveCfg = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Any CPU.Build.0 = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Win32.ActiveCfg = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Debug|Win32.Build.0 = Debug|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Release|Any CPU.ActiveCfg = Release|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Release|Mixed Platforms.ActiveCfg = Release|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Release|Mixed Platforms.Build.0 = Release|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Release|Win32.ActiveCfg = Release|Win32

{A3B39452-3ED5-4CF6-A893-52F36999C458}.Release|Win32.Build.0 = Release|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Any CPU.ActiveCfg = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Any CPU.Build.0 = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Win32.ActiveCfg = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Debug|Win32.Build.0 = Debug|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Release|Any CPU.ActiveCfg = Release|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Release|Mixed Platforms.ActiveCfg = Release|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Release|Mixed Platforms.Build.0 = Release|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Release|Win32.ActiveCfg = Release|Win32

{D117E04B-E566-4E6D-8DB4-A51B80BA6E56}.Release|Win32.Build.0 = Release|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Any CPU.ActiveCfg = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Any CPU.Build.0 = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Win32.ActiveCfg = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Debug|Win32.Build.0 = Debug|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Release|Any CPU.ActiveCfg = Release|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Release|Mixed Platforms.ActiveCfg = Release|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Release|Mixed Platforms.Build.0 = Release|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Release|Win32.ActiveCfg = Release|Win32

{F089E784-C194-4208-A2C6-58879F188406}.Release|Win32.Build.0 = Release|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Any CPU.ActiveCfg = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Any CPU.Build.0 = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Win32.ActiveCfg = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Debug|Win32.Build.0 = Debug|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Release|Any CPU.ActiveCfg = Release|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Release|Mixed Platforms.ActiveCfg = Release|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Release|Mixed Platforms.Build.0 = Release|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Release|Win32.ActiveCfg = Release|Win32

{4821031C-D418-462B-8B51-78BFD0CD2163}.Release|Win32.Build.0 = Release|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Any CPU.ActiveCfg = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Any CPU.Build.0 = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Win32.ActiveCfg = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Debug|Win32.Build.0 = Debug|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Release|Any CPU.ActiveCfg = Release|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Release|Mixed Platforms.ActiveCfg = Release|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Release|Mixed Platforms.Build.0 = Release|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Release|Win32.ActiveCfg = Release|Win32

{E58755B0-8E17-4B29-818D-99077002FA98}.Release|Win32.Build.0 = Release|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Any CPU.ActiveCfg = Debug|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Any CPU.Build.0 = Debug|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Mixed Platforms.Build.0 = Debug|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Win32.ActiveCfg = Debug|Win32

{EC883BCB-9D97-4709-B67D-62DC7405C728}.Debug|Win32.Build.0 = Debug|Win32
{EC883BCB-9D97-4709-B67D-62DC7405C728}.Release|Any CPU.ActiveCfg = Release|Win32
{EC883BCB-9D97-4709-B67D-62DC7405C728}.Release|Mixed Platforms.ActiveCfg = Release|Win32
{EC883BCB-9D97-4709-B67D-62DC7405C728}.Release|Mixed Platforms.Build.0 = Release|Win32
{EC883BCB-9D97-4709-B67D-62DC7405C728}.Release|Win32.ActiveCfg = Release|Win32
{EC883BCB-9D97-4709-B67D-62DC7405C728}.Release|Win32.Build.0 = Release|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Any CPU.ActiveCfg = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Any CPU.Build.0 = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Mixed Platforms.Build.0 = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Win32.ActiveCfg = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Debug|Win32.Build.0 = Debug|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Release|Any CPU.ActiveCfg = Release|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Release|Mixed Platforms.ActiveCfg = Release|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Release|Mixed Platforms.Build.0 = Release|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Release|Win32.ActiveCfg = Release|Win32
{CBCE8A17-0CF7-45EF-9DDE-D57CB5AD4F35}.Release|Win32.Build.0 = Release|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Any CPU.ActiveCfg = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Any CPU.Build.0 = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Mixed Platforms.Build.0 = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Win32.ActiveCfg = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Debug|Win32.Build.0 = Debug|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Release|Any CPU.ActiveCfg = Release|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Release|Mixed Platforms.ActiveCfg = Release|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Release|Mixed Platforms.Build.0 = Release|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Release|Win32.ActiveCfg = Release|Win32
{4BF2D291-5A9C-41BE-9DDC-272BB37E6A83}.Release|Win32.Build.0 = Release|Win32
{0F7AD900-FCCE-40CF-942D-A6EB769432E7}.Debug|Any CPU.ActiveCfg = Debug|Win32
{0F7AD900-FCCE-40CF-942D-A6EB769432E7}.Debug|Any CPU.Build.0 = Debug|Win32
{0F7AD900-FCCE-40CF-942D-

A6EB769432E7}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

A6EB769432E7}.Debug|Mixed Platforms.Build.0 = Debug|Win32

A6EB769432E7}.Debug|Win32.ActiveCfg = Debug|Win32

A6EB769432E7}.Debug|Win32.Build.0 = Debug|Win32

A6EB769432E7}.Release|Any CPU.ActiveCfg = Release|Win32

A6EB769432E7}.Release|Mixed Platforms.ActiveCfg = Release|Win32

A6EB769432E7}.Release|Mixed Platforms.Build.0 = Release|Win32

A6EB769432E7}.Release|Win32.ActiveCfg = Release|Win32

A6EB769432E7}.Release|Win32.Build.0 = Release|Win32

28619EB34D02}.Debug|Any CPU.ActiveCfg = Debug|Win32

28619EB34D02}.Debug|Any CPU.Build.0 = Debug|Win32

28619EB34D02}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

28619EB34D02}.Debug|Mixed Platforms.Build.0 = Debug|Win32

28619EB34D02}.Debug|Win32.ActiveCfg = Debug|Win32

28619EB34D02}.Debug|Win32.Build.0 = Debug|Win32

28619EB34D02}.Release|Any CPU.ActiveCfg = Release|Win32

28619EB34D02}.Release|Mixed Platforms.ActiveCfg = Release|Win32

28619EB34D02}.Release|Mixed Platforms.Build.0 = Release|Win32

28619EB34D02}.Release|Win32.ActiveCfg = Release|Win32

28619EB34D02}.Release|Win32.Build.0 = Release|Win32

3C7084883355}.Debug|Any CPU.ActiveCfg = Debug|Win32

3C7084883355}.Debug|Any CPU.Build.0 = Debug|Win32

3C7084883355}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32

3C7084883355}.Debug|Mixed Platforms.Build.0 = Debug|Win32

3C7084883355}.Debug|Win32.ActiveCfg = Debug|Win32

3C7084883355}.Debug|Win32.Build.0 = Debug|Win32

3C7084883355}.Release|Any CPU.ActiveCfg = Release|Win32

3C7084883355}.Release|Mixed Platforms.ActiveCfg = Release|Win32

3C7084883355}.Release|Mixed Platforms.Build.0 = Release|Win32

3C7084883355}.Release|Win32.ActiveCfg = Release|Win32

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{0F7AD900-FCCE-40CF-942D-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{BD820F82-1905-4463-9A61-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

{92161C45-0F41-4688-A4EE-

```
        3C7084883355}.Release|Win32.Build.0 = Release|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Any CPU.ActiveCfg = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Any CPU.Build.0 = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Mixed Platforms.ActiveCfg = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Mixed Platforms.Build.0 = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Win32.ActiveCfg = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Debug|Win32.Build.0 = Debug|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Release|Any CPU.ActiveCfg = Release|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Release|Mixed Platforms.ActiveCfg = Release|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Release|Mixed Platforms.Build.0 = Release|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Release|Win32.ActiveCfg = Release|Win32
                                                {D65CA7DC-7FAF-4C43-BB82-
62F4EB0ED55F}.Release|Win32.Build.0 = Release|Win32
                                        EndGlobalSection
                                GlobalSection(SolutionProperties) = preSolution
                                                HideSolutionNode = FALSE
                                        EndGlobalSection
        EndGlobal
```