Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«МЕТОД ТА ЗАСІБ ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ»

Виконав: студент 2-го курсу, групи 1БС-20м
спеціальності 125 – Кібербезпека
(шифр і назва напряму підготовки, спеціальності)

_____ Ціхоцький М.С.
(прізвище та ініціали)

Керівник к.т.н., доц. каф. ЗІ
_____ Баришев Ю.В.
(прізвище та ініціали)

Опонент: к.т.н., доц. каф . ОТ
_____ Савицька Л. А.
(прізвище та ініціали)

«__» _____ 2021 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____ Лужецький В. А.
«___» _____ 2021 р.

Вінниця ВНТУ – 2021 рік

АНОТАЦІЯ

УДК 004.056.55

Ціхоцький М.С. Метод та засіб псевдонедетермінованого блокового шифрування файлів. Магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2021. 97 с.

Укр. мовою. Бібліогр.: 34 назв; рис.: 21; табл. 11.

Магістерська кваліфікаційна робота присвячена розробці методу та засобу псевдонедетермінованого блокового шифрування файлів. Підготовлено науково-дослідне та техніко-економічне обґрунтування доцільності проведення досліджень. Проведено аналіз сучасних методів блокового шифрування, виходячи з результатів було запропоновано підхід для розробки нового блокового шифру. Розроблено програмний засіб, який виконує операції зашифрування та розшифрування файлів з використанням запропонованого методу. Виконано тестування засобу псевдонедетермінованого блокового шифрування файлів.

Ілюстративна частина складається з 7 плакатів з демонстрацією результатів моделювання і проведених досліджень.

Оцінку витрат на розробку представлено в економічному розділі.

Ключові слова: симетричне шифрування, блоковий шифр, метод шифрування, секретний ключ, криптостійкість.

ABSTRACT

Tsikhotskyi M.S. Method and means of pseudo-nondeterministic block file encryption. Master's thesis in specialty 125 – cybersecurity. educational program - Security of information and communication systems. Vinnytsia: VNTU, 2021. 97 p.

In Ukrainian language. Bibliographer: 33 titles; fig.: 21; table. 11.

Master's degree qualification work is devoted to the development of a method and means of pseudo-nondeterministic block file encryption. Scientific and technical rationale for the feasibility of research were prepared. The analysis of modern methods of block encryption is carried out. Based on the results, an approach to the development of a new block cipher was proposed. A software tool was developed that performs operations on file encryption and decryption using the proposed method. The pseudo-nondeterministic block file encryption tool was tested.

The graphic part consists of 7 posters demonstrating the results of modeling and research.

The cost estimate for the development of the method is presented in the economic section.

Keywords: symmetric encryption, block cipher, encryption method, secret key, crypto infeasibility.

ЗМІСТ

ВСТУП

У наш час, коли діяльність людини все більше переноситься у дистанційний режим, а отже разом з цим виникає необхідність передачі різного виду інформації. Сьогодні існує безліч різних варіантів для передавання даних, для прикладу – текстові повідомлення, аудіо-записи, відео-повідомлення, файли, листи та інші. Для кожного з видів трансляції інформації визначається певний рівень її значущості.

Інформація є ключовим ресурсом у всіх сферах людської діяльності, отже, захист інформації відіграє дуже важливу роль у сучасному світі. Безліч даних зберігаються на різноманітних ресурсах як у мережі, так і на фізичних носіях. Інформація може бути розглянута під різними кутами, але базовими для безпеки [1] принципами є конфіденційність, цілісність та доступність [2]. При цьому переважна більшість людей не звертають увагу на способи захисту під час зберігання інформації (наприклад файлів, оскільки у більшості випадків всі дані частіше всього знаходяться саме у цьому форматі). Аби забезпечити окремий аспект безпеки інформації або навіть усі разом було розроблено низку механізмів (протоколів), які регулюють та сприяють безпечній обробці даних.

Для захисту файлів доречно використовувати такий вид криптографічних перетворень як блокові шифри. За цей час багато алгоритмів шифрування було добре досліджено та знайдено вразливості до різних видів атак. Чим старішим є алгоритм шифрування тим більшою є ймовірність того що будуть виявлені вразливості до певного виду досліджень.

Актуальність пошуку нових, більш стійких методів блокового шифрування є затребуваною виходячи з аналізу сучасних стандартів блокового шифрування, які в більшості випадків є морально застарілими та мають певні вразливості до різних методів криптоаналізу.

Об’єктом дослідження є процес забезпечення надійного шифрування файлів.

Предметом дослідження є методи блокового шифрування, а також процес псевдонедетермінованості, як підхід до шифрування.

Метою магістерської кваліфікаційної роботи є покращення існуючих підходів до шифрування файлів.

Засіб, який буде отримано в результаті виконання магістерської роботи можливо буде використовувати у програмних застосунках як механізм забезпечення аспекту конфіденційності при передачі файлів довільної довжини.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

– проаналізувати відомі методи блокового шифрування;
– проаналізувати режими роботи блокових шифрів;
– виконати узагальнений математичний опис процесу шифрування;
– розробити метод псевдонедетермінованого блокового шифрування;
– розробити алгоритм роботи програмного засобу;
– реалізувати програмний засіб для шифрування файлів на основі розробленого методу;
– провести тестування методу шифрування;
– виконати розрахунки економічної доцільності розробки.

Наукова новизна магістерської роботи полягає в тому, що було запропоновано удосконалений метод шифрування файлів з використанням процесу псевдонедетермінованості, який на відміну від відомих передбачає внесення невизначеності в спосіб організації раундового перетворення, що дозволяє підвищити рівень стійкості шифротексту до розкриття сторонніми особами.

Практична цінність: метод псевдонедетермінованого блокового шифрування файлів; програмний засіб шифрування файлів.

Результати магістерської роботи доповідалися на XLIX науково-технічній конференції факультету інформаційних технологій та комп'ютерної інженерії.

1 АНАЛІЗ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1.1 Науково-дослідне обґрунтування

Сьогодні питання захисту інформації є одним за найважливіших завдань інформаційної безпеки, оскільки, отримавши доступ до певних даних, зловмисник може маніпулювати ними для створення критичних ситуацій для власників чиї дані були скомпрометовані. Що в свою чергу може призвести до отримання збитків різного масштабу, в залежності від рівня важливості даної інформації. Для захисту інформації існує сукупність методів та засобів, які забезпечують цілісність, конфіденційність та доступність від впливу на неї різноманітних загроз. Виділяють такі види захисту – технічний, інженерний, криптографічний та організаційний [3]. У даній роботі розглядається криптографічний напрям для захисту даних.

Симетрична криптографія (відома також як криптографія із секретним ключем) — це використання єдиного спільного секрету для обміну зашифрованими даними між сторонами [4]. Шифри в цій категорії називаються симетричними, оскільки використовується один і той же ключ для шифрування та розшифрування даних.

Симетричне шифрування є двостороннім процесом. З блоком відкритого тексту і заданим ключем симетричні шифри завжди створюватимуть один і той самий шифротекст. Так само, використання такого ж ключа на відповідному блоці зашифрованого тексту завжди створюватиме ідентичний вихідний відкритий текст. Симетричне шифрування корисне для захисту даних між сторонами за допомогою вбудованого спільного ключа, а також часто використовується для зберігання конфіденційних даних.

Симетричне шифрування ділиться на два напрямки: потокове шифрування та блокове шифрування[5].

Оскільки більшість інформації зберігається у форматі файлів, для дослідження було обрано саме блокове шифрування, яке є більш доцільним для захисту даних великого обсягу. Також блокове шифрування є більш

дослідженим, що, в свою чергу, створює умови як для появи вразливостей у існуючих підходах до шифрування, так і для розвитку більш стійких блокових шифрів на основі наявної бази інформації.

Для синтезу нового підходу в реалізації методу блокового шифрування потрібно розглянути існуючі сучасні підходи та методи шифрування, що дасть змогу визначити найбільш актуальний напрям для проведення подальшого дослідження.

1.2 Аналіз відомих методів блокового шифрування

Сучасний блоковий шифр – це шифр з симетричним ключем, який розділяє перед шифруванням відкритий текст на n – бітні блоки і далі шифрує повідомлення блоками.

$$y = E_k(x) \qquad \text{та} \qquad x = D_k(y),$$

де x, y – блоки відкритого та зашифрованого текстів; k – секретний ключ шифру; E, D – функції шифрування та розшифрування.

Алгоритм розшифрування та зашифрування – інверсні, обидва працюють на одному й тому ж самому секретному ключі. Алгоритм процесів шифрування/розшифрування зображений на рисунку 1.1.

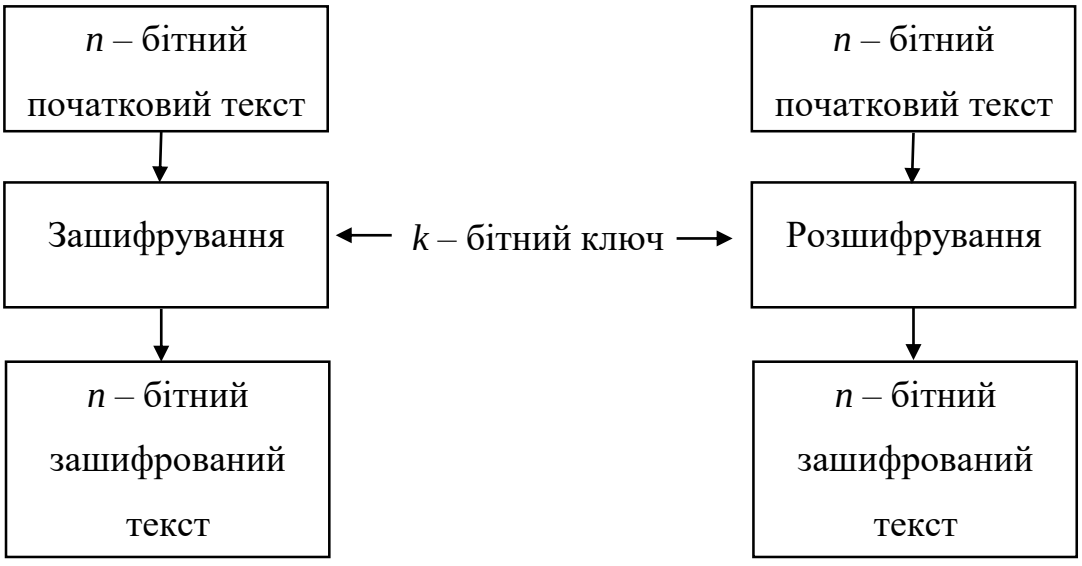


Рисунок 1.1 – Загальна схема шифрування та розшифрування за допомогою блокового шифру

Якщо повідомлення містить менше *n* біт, тоді його доповнюють додатковими даними (частіше всього нулями) до *n* – бітного розміру; якщо текст містить більше, ніж *n* біт, то його ділять на *n* – бітні блоки. Найчастіше зустрічаються блокові шифри, які оперують блоками довжиною *n = 64, 128, 256, 512* біт[6].

Згідно до робіт Шеннона[7] від сучасних блокових шифрів вимагають такі вимоги:

– розсіювання інформації – поширення впливу одного знаку відкритого тексту на майже всі знаки шифротексту. Це ускладнює статистичний аналіз та не дозволяє відновити ключ по частинам;
– Перемішування інформації для ускладнення залежності між ключем та зашифрованих текстом.

Схеми, які виконують такі задачі, називають SP-мережами. Основні оператори блокових шифрів:

– оператори перестановки, так звані *P* – бокси;
– оператори підстановки, так звані *S* – бокси;
– операція виключної диз'юнкції (XOR);
– циклічний зсув;
– заміна;
– розбиття та об’єднання блоків.

Комбінація цих операторів в сучасних шифрах дозволяє розсіяти та перемішати інформацію

P – бокс (блок перестановки) схожий на традиційну перестановку (символи, які переставляються – біти). Існують 3 типи *P* – боксів:

1) прямі *P* – бокси, структура даного типу зображена на рисунку 1.2;

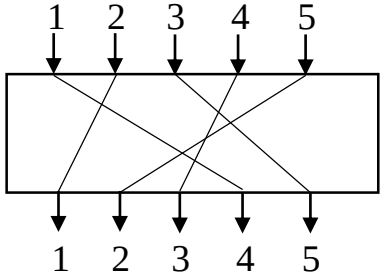


Рисунок 1.2 – Прямий *P*–бокс 5х5

2) *P* – бокс ущільнення, структура даного типу зображена на рисунку 1.3;

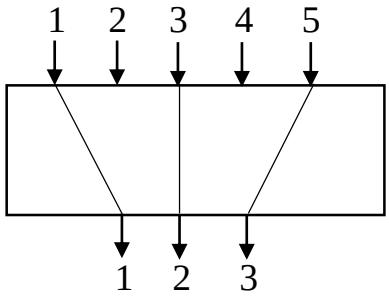


Рисунок 1.3 – *P*–бокс ущільнення 5х3

3) Р – бокс розширення, структура даного типу зображена на рисунку 1.4.

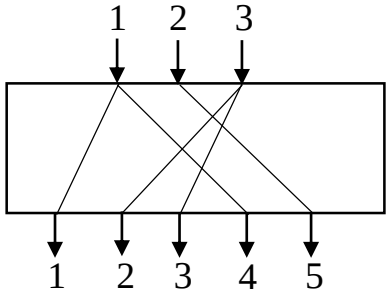


Рисунок 1.4 – *P*–бокс розширення 3х5

Прямий *P*-бокс зазвичай задають за допомогою таблиці перестановок, в якій номери клітинок визначають номери бітів вхідного блоку, а числа записані в клітинки вказують нові позиції біта. Прямий *P*-бокс – зворотній, тобто його можна використати для шифрування та розшифрування.

P-бокс ущільнення – це *P*-бокс з *n*-входами та *m*-виходами, де $m < n$. Деякі з інформаційних входів заблоковані і не зв’язані з виходом. Задаються таблицею перестановки з *m* клітинками (входи), в яких позиції бітів від1 до *n*.

P-бокс розширення – це *P*-бокс з *n*-входами та *m*-виходами, де $m > n$. Деякі з входів зв’язані з більше чим одним виходом.

P-бокси ущільнення та розширення незворотні. *P*-бокс стискання не є зворотнім до *P*-боксу розширення. Це означає, що якщо використовуються *P*-бокс ущільнення для шифрування, то ми не можливо використовувати *P*-бокс розширення для розшифрування і навпаки. Хоча є шифри, які використовують *P*-бокси стискання та розширення.

S-бокс (блок підстановок) – це мініатюрний шифр підстановки. На вхід в *S*-бокс може подаватись *n*-бітний блок, а виході отримують вже *m*-бітний блок, при чому не завжди $n = m$.

Нехай в *S*-боксі з n входами та m виходами входи позначаються як $x_1, x_2, \dots, x_n$, а виходи – як $y_1, y_2, \dots, y_m$. Зв'язок між входами там виходами визначає система рівнянь:

$$\begin{cases} y_1 = f_1(x_1, x_2, \dots, x_n), \\ y_2 = f_1(x_1, x_2, \dots, x_n) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ y_m = f_m(x_1, x_2, \dots, x_n) \end{cases}$$

S-бокси поділяються на лінійні та нелінійні. В лінійному *S*-боксі зв'язок можна описати в виді лінійних співвідношень:

$$\begin{cases} y_1 = a_{11}x_1 \oplus a_{12}x_2 \oplus \dots \oplus a_{1n}x_n , \\ y_2 = a_{21}x_1 \oplus a_{22}x_2 \oplus \dots \oplus a_{2n}x_n , \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ y_m = a_{m1}x_1 \oplus a_{m2}x_2 \oplus \dots \oplus a_{mn}x_n \end{cases}$$

В нелінійному *S*-боксі лінійні співвідношення для кожного входу вказати неможливо.

Далі доцільно розглянути операції, які часто використовуються при побудові блокових шифрів.

Операція XOR – побітове множення за модулем 2. Властивості операції XOR:

- Замкненість; якщо *x, y* – *n*-бітні слова, то $x \oplus y = z$, де *z* – *n*-бітне слово*;*
- Асоціативність: $x \oplus (y \oplus z) = (x \oplus (y) \oplus z)$;
- Комутативність: $x \oplus y = y \oplus x$;
- Існування нульового елементу при умові $x \oplus (00 \dots 0) = x$;
- Існування інверсії – операція XOR слова самого з собою дає нульовий елемент x$\oplus x = (00 \dots 0)$;

Циклічний зсув – циклічний правий зсув, який зсуває кожен біт у *n*-бітовому слові на *k* позицій праворуч; самі праві *k*-біт видаляються і стають крайніми лівими. Аналогічно працює лівий зсув. Циклічний лівий зсув – є

результатом інверсії правого зсуву. Якщо один з них використовується для шифрування, інший може використовуватись для розшифрування.

Заміна – частковий випадок операції циклічного зсуву на $k = n/2$ бітів (тут n-парне). Зсув ліворуч на $n/2$ – це те саме, що зсув на $n/2$ вправо, операція є оборотною.

Розбиття розділяє n-бітне слово навпіл, створюючи два слова рівної довжини. Об'єднання зв'язує два слова рівної довжини, для того, щоб створити n-бітне слово. Ці дві операції інверсні одна до одної.

Під час виконання цієї роботи було досліджено 12 найбільш використовуваних алгоритмів блокового шифрування на предмет основних параметрів блокового шифру, таких як розмір блоку шифрування та розмір ключа. Також визначено тип реалізації кожного методу SW – програмний, HW – апаратний. Основним напрямком у пошуках були відомості про те чи були успішні спроби зламати той чи інший алгоритм. Результати проведення цього дослідження наведено в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика відомих алгоритмів блокового шифрування

Алгоритм шифрування	Розмір блоку [біт]	Ключ [біт]	Імплементація	Захищеність
AES[8]	128	128, 192, 256	SW, HW	Стійкий
Kalyna[9]	128	128, 192, 256	SW, HW	Стійкий
Camellia[10]	128	128, 192, 256	SW, HW	Стійкий
GOST[11]	64	256	HW	Зламаний
IDEA[12]	64	128	SW	Відносно стійкий
KASUMI[13]	64	128	SW, HW	Зламаний
KATAN[14]	32, 48, 64	128	HW	Відносно стійкий

Продовження таблиці 1.1.

Алгоритм шифрування	Розмір блоку [біт]	Ключ [біт]	Імплементація	Захищеність
Noekeon[15]	128	128, 256	SW, HW	Відносно стійкий
Serpent[16]	128	128, 192, 256	SW, HW	Стійкий
Skipjack[17]	64	80	SW, HW	Зламаний
Twofish[18]	128	128, 192, 256	SW, HW	Стійкий
XTEA[19]	64	128	SW, HW	Стійкий

Для певних різновидів AES, які використовуються для шифрування сертифікатів безпеки у мережі, було знайдено вразливості до атаки на час кешування, при яких є можливість видобути секретний ключ, через вразливості у S-box компонентах. У режимі шифрування AES-ECB для всіх нульових блоків, виконується шифрування з однаковим значенням ключа зберігається в змінній, яка називається *H*, така поведінка значно спрощує зловмисникам аналіз шифротексту. Шифр Kalyna, підлягає атаці на бічні канали при процесі розгортання ключа. Майже кожна мікро-архітектурна функція сучасних процесорів може використовуватися для атаки криптографічних примітивів через апаратні межі безпеки. Також були реалізовані атаки на шифр IDEA, до зменшеної кількості раундів.

Під час аналізу приведених вище блокових шифрів було помічено, що у кожного з них є певні недоліки та на частину можливо виконати атаки (навіть теоретично, але наразі невідомо як швидко теоретична можливість зміниться на практичну). Тому було вирішено запропонувати новий підхід для виконання шифрування, який зможе забезпечити більшу стійкість з оглядом на швидкий темп розвитку технологій.

1.3 Аналіз режимів роботи блокових шифрів

Для виконання самого процесу шифрування та розшифрування даних існують різні види режимів роботи блокових шифрів. Усі затверджені режими є описані у NIST (National Institute of Standards and Technology).

Для вибору найбільш вдалого режиму шифрування було розглянуто чотири основні – це ECB, CBC, CTR, GCM. Основна увага при виконанні аналізу була направлена простоту реалізації та високий рівень захисту до відомих атак.

Електронна кодова книжка (ECB) – це найпростіший режим. Певний обсяг даних шифрується так, що кожен раунд може бути виконаний незалежно від інших з використанням секретного ключа.

Перевага цього методу – це швидкість виконання процесу шифрування та розшифрування. Окремі блоки є незалежними один від одного. Недоліком є те, що однаковий вхідний текстовий блок завжди буде зашифрований однаково. І якщо буде знайдено декілька однакових блоків шифротексту, вхідний текст може буде відновлений.

Шифрування виконується за таким законом;

$$\mathrm{C}_i = E_k(P_i),$$

де $i = 0, \dots, 3$.

Лічильник (CTR) – даний режим перетворює блоки шифру в синхронізований поточний шифр. Вектор ініціалізації включається в лічильник на початку шифрування, як результат – шифротекст. Далі лічильник оновлюється (частіше за все додаванням одиниці до поточного стану) та отримується наступний блок шифротексту. Основний принцип полягає в тому що один і той самий блок даних не може бути зашифрований тим самим значенням ключа. Процес шифрування та розшифрування виглядає таким чином:

$$CTR_i = |IV + i - 1|;\ H_i = E_k(CTR_i);\ C_i = P_i \oplus H_i;$$
$$CTR_i = |IV + i - 1|;\ H_i = E_k(CTR_i);\ P_i = C_i \oplus H_i,$$

де i = 1, 2, …, n.

Ланцюговий блоковий шифр (CBC) режим представляє залежність кожного зашифрованого блоку. Залежність досягається за рахунок виконання операцію XOR над наступним блоком даних та попереднім значенням шифротексту. Це, означає що повідомлення шифрується не тільки власне значення та секретний ключ, а також з використанням попереднього блоку. Перший блок є вектором ініціалізації, котрий є випадковим значенням. Вектор забезпечує умову того що результат шифрування двох однакових повідомлень буде різним. Якщо буде використано некоректний вектор з вірним ключем, всі блоки будуть розшифровано некоректно.

Процес шифрування можна описати наступним чином:

$$C_0 = IV, C_i = E_k(P_i \oplus C_{i-1})$$

Процес розшифрування можна описати наступним чином:

$$C_0 = IV, P_i = D_k(C_i) \oplus C_{i-1}$$

Режим лічильника Галуа (GCM) – це керований режим шифрування який забезпечує конфіденційність та цілісність даних. Цей режим є рекомендований для високо-швидкісних передач даних, тому що шифрування виконується в режимі лічильника, який перетворює блоковий шифр в потоковий шифр. GSM також використовує геш-функцію, яка позбавляє головного недоліку – неможливість апаратної реалізації. Даний режим дещо відрізняється від попередніх, оскільки використовує додаткові дані для забезпечення авторизованого шифрування інформації.

Процес шифрування потребує 4 вхідних множини:

– секретний ключ (K);
– вектор ініціалізації (IV);
– вхідний текст (P);
– додаткові автентичні дані (A).

Результати проведеного аналізу режимів шифрування наведено у таблиці 1.2.

Таблиця 1.2 – Порівняння режимів блокового шифрування

Режим	Переваги	Недоліки
ECB	Висока швидкість; паралельність; простий.	Немає перевірки цілісності
CBC	Властивість множення помилок; Ланцюговий; Поширений; Паралельність, але тільки для розшифрування.	Залежність від попереднього блоку; Повільний; Не може бути паралельний при шифруванні. Потрібно додатково реалізовувати процес розшифрування.
CTR	Паралельність; Поточне шифрування; Довжина повідомлення може бути довільною; Не потрібно додатково реалізовувати процес розшифрування.	Відсутність властивості множення помилок.
GCM	Конфіденційність; Цілісність; Висока швидкість; Паралельність.	Складність; Потужність обчислення.

Виходячи з проведеного аналізу основних режимів блокового шифрування. Було вирішено, що режим EBC та CBC є недоцільними через свої недоліки. Тому для розробки методу блокового шифрування буде доцільно використати GCM або CTR режим, в залежності від конкретних вимог.

1.4 Економічне обґрунтування доцільності розробки

Для визначення економічної доцільності розробки було проаналізовано сучасний стан, щодо різних варіантів програмних рішень для шифрування

файлів. Оцінено якісні та економічні характеристики засобів в порівнянні з запропонованим методом псевдонедетермінованого блокового шифрування. За результатами оцінки можливо прогнозувати більш економічно та функціонально вигідний варіант використання запропонованого методу шифрування. Також було визначено, що продукт має високий рівень самоокупності за відносно невеликий проміжок часу.

1.5 Постановка задачі дослідження

Інформація є дуже важливим ресурсом у сучасному світі. З поточним прискореним переходом більшості працівників на віддалений режим роботи, є особлива необхідність більш пильного керування процесами обміну та збереження даних. З кожним днем обсяг електронного документообігу все зростає.

Для забезпечення надійного зберігання та передачі такого виду інформації як файли, є доцільним використання є процес блокового шифрування. Особливу увагу варто звернути на вибір засобу (алгоритму) для виконання шифрування/розшифрування. Так деякі з доволі популярних методів шифрування мають низку вразливостей, що може спричинити до небажаного витоку конфіденційної інформації та спричинити втрати.

Аналізуючи найбільш поширені методи шифрування можна дійти висновку, що більшість мають доволі застарілу архітектуру та мають теоретично доведену стійкість. Отже, в один момент будь-хто може віднайти вразливість у алгоритмі, після чого всі системи, в яких використовувався даний метод шифрування стануть вразливими.

Для запобігання виникнення такої ситуації є необхідність у дослідженні та розробці нових методів блокового шифрування. Як один з підходів було запропоновано використати сталі схеми, на яких будується більшість блокових шифрів та додати процес псевдонедетермінованості, який відповідає за визначення конкретної схеми на основі якої буде виконуватись ітерація шифрування.

Такий підхід дозволяє підвищити стійкість до лінійного та диференційного крипто аналізу. А також його перевага полягає в тому що, якщо в одній з використовуваних схем буде знайдена вразливість, інша частина алгоритму шифрування залишиться стійкою і зловмисники не будуть мати змоги швидко виконати розшифрування файлів.

Для розробленого методу псевдонедетермінованого блокового шифрування потрібно виконати математичний опис множин та процесів. Виконати розробку алгоритмів роботи методу та засобу шифрування. Розроблені алгоритми реалізувати у вигляді програмного засобу, який буде виконувати основні функції криптографічного перетворення, а саме шифрування та розшифрування файлів.

На основі сформованих вимог до засобу псевдонедетермінованого блокового шифрування файлів розроблено технічне завдання.

1.6 Висновки до розділу

Отже, у даному розділі було наведено важливість захисту файлів як виду інформації. Захист файлів за допомогою методів блокового шифрування дозволяє мати доступ до зашифрованих даних лише користувачам, які знають секретний ключ.

Досліджено сучасні методи блокового шифрування, визначено ключові структури на яких вони побудовані, переваги та недоліки кожного з розглянутих методів. Результатом є таблиця порівняння різних методів блокового шифрування.

Також досліджено чотири основних режими роботи блокових шифрів, наведено порівняльну таблицю, де було вказано переваги та недоліки кожного.

Обрано два режими шифрування, які задовольняють сучасні вимоги відповідно до низки стандартів, для подальшого використанні у розробці методу шифрування.

На основі отриманих результатів аналізу доцільно перейти до математичного опису методу шифрування та його подальшого дослідження.

2 МЕТОД ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ

2.1 Математичний опис процесу шифрування

За використанням теоретико-множинного підходу виконано формалізований опис процесу блокового шифрування, що дасть змогу узагальнити підходи до шифрування. Структура шифру розглядається як сукупність взаємопов'язаних процесів, що в свою чергу є елементами криптографічних перетворень.

Для початку необхідно описати загальну структуру виконання процесу блокового шифрування. Представимо процес шифрування як набір множин відкритого тексту – plainText, шифротексту – cipherText, секретного ключа – secretKey та криптографічних перетворень – transformations:

Block_Encryption = {plainText, cipherText, secretKey, transformations},

де plainText = {0, 1, ..., 2^{n_b}-1}, де n_b — розрядність блоку даних шифру;
cipherText = {0, 1, ..., 2^{n_b}-1}, де n_b — розрядність блоку даних шифру;
secretKey = {0, 1, ..., 2^{n_k}-1}, де n_k — розрядність ключа шифрування;
transformations = {enc(k , p), dec(k , c)}, де p ∈ plainText; c ∈ cipherText.

За допомогою даного математичного опису можливо описати більшість розглянутих сучасних методів блокового шифрування.

Далі доцільно описати запропонований новий метод псевдонедетермінованого блокового шифру, який містить відмінності від загальної структури:

PND_Block_Encryption = {plainText, cipherText, secretKey, PND_Transformations, operationVector}.

У даному методі було додано множину PND_Transformations, яка містить розширений набір елементів з множин enc(·), dec(·), оскільки вони в

свою чергу використовують дві структури шифрування. Та operationVector – множина значень, яка відповідає за процес вибору поточної структури для виконання процесу шифрування на кожному раунді та отримується розподілом секретного ключа:

$$PNDtransformations = \{\{enc(k, p)\}, \{dec(k, c)\}, derivation(\cdot)\},$$

де derivation = key = k || v_0

Для фіксованого набору вхідних даних (повідомлення або файлу) на кожній і-й ітерації використовуються процедури $\{enc\_k_{v_i(\cdot)}, dec\_k_{v_i(\cdot)}\}$ зашифрування та розшифрування відповідно, де $v_i \in$ operationVector.

Для того, щоб оцінити ефективність запропонованого методу, пропонується розглянути такий клас атак на криптографічні перетворення як диференційний аналіз. Основна ідея проведення аналізу полягає в збиранні статистики, щодо змін вхідних та вихідних бітів.

Якщо розглядати загальну структуру виконання процесу блокового шифрування, то припустимо, для щоб видобути корисну інформації на основі проведення диференційного аналізу зловмиснику необхідно зібрати *n*-пар значень вхід/вихід:

Diff_analysis(Block_Encryption) = $\{n_{plainText}, n_{cipherText}\}$,

де $n_{plainText}$ – кількість отриманих вхідних значень;

$n_{cipherText}$ – кількість отриманих вихідних значень.

Тоді як при використанні запропонованого методу потрібно буде додатково ще вірно інтерпретувати кожний раунд шифрування. Тобто пара *plainText/cipherText,* може бути отримана на основі виконання двох різних криптографічних перетворень, що відповідно збільшує складність до виконання диференційного криптоаналізу в два рази.

$$Diff\_analysis(PND\_Block\_Encryption) = 2 * \{n_{plainText}, n_{cipherText}\}$$

Отже, було представлено математичний опис псевдонедетермінованого блокового шифрування та наведено додаткові множини, які відрізняються від загальних структур процесу шифрування.

Далі, доцільно буде розглянути різні види конструкцій для шифрування.

2.2 Конструкції блокового шифрування

Більшість сучасних блокових шифрів – це так звані продуктові шифри. Продуктові шифри поєднують два або більше перетворення для того щоб у поєднанні вони робили шифрування більш стійким[20]. Зазвичай шифри даного виду містять такі перетворення як *S*-бокси, *P*-бокси та модульну арифметику. Такий підхід був запропонований Шенноном[21], та узагальнений Х. Фейстелем для того щоб задовольнити лавинний ефект під час шифрування[22].

Суть цього ефекту наступна – якщо хоч трохи змінити дані на вході блокового шифру (текст повідомлення, секретний ключ), то вихідні дані будуть різко відрізнятись. Даний ефект є затребуваною властивістю для шифрів, оскільки якщо розроблений шифр задовольняє його, стійкість шифрування значно збільшується. Для прикладу, суворий лавинний критерій (strict avalanche criterion) є сильною версією вимоги щодо якісних лавинних властивостей шифру. Доповнення до будь-якого окремого біта або ключа повинна давати рівно 50% ймовірності зміни будь-якого біта даних на виході[23]. Лавинного ефекту можливо досягти за якісного поєднання криптографічних перетворень (які були розглянуті у пункті 1.2) та раундової функції. Зокрема виділяються декілька найбільш відомих криптографічних структур, які використовуються у сучасній криптографії.

Шифр Фейстеля — це структура, запропонована Хорстом Фейстелем, яка була використана при розробці багатьох симетричних блочних шифрів. Власне, структура, запропонована Фейстелем, заснована на структурі Шеннона, яка була запропонована в 1945 році [24]. Структура Шеннона пропонує реалізацію перемішування та дифузії по черзі. Перемішування

створює складне відношення між зашифрованим текстом і ключем шифрування шляхом реалізації складного алгоритму заміни. Тоді як дифузія створює складне відношення між відкритим текстом і зашифрованим текстом, реалізуючи більш складний алгоритм перестановки.

Шифр Фейстеля запропонував структуру, яка по черзі реалізує підстановку і перестановку. Підстановка здійснюється шляхом заміни елементів відкритого тексту або набору елементів відкритого тексту елементом зашифрованого тексту або набором елементів зашифрованого тексту.

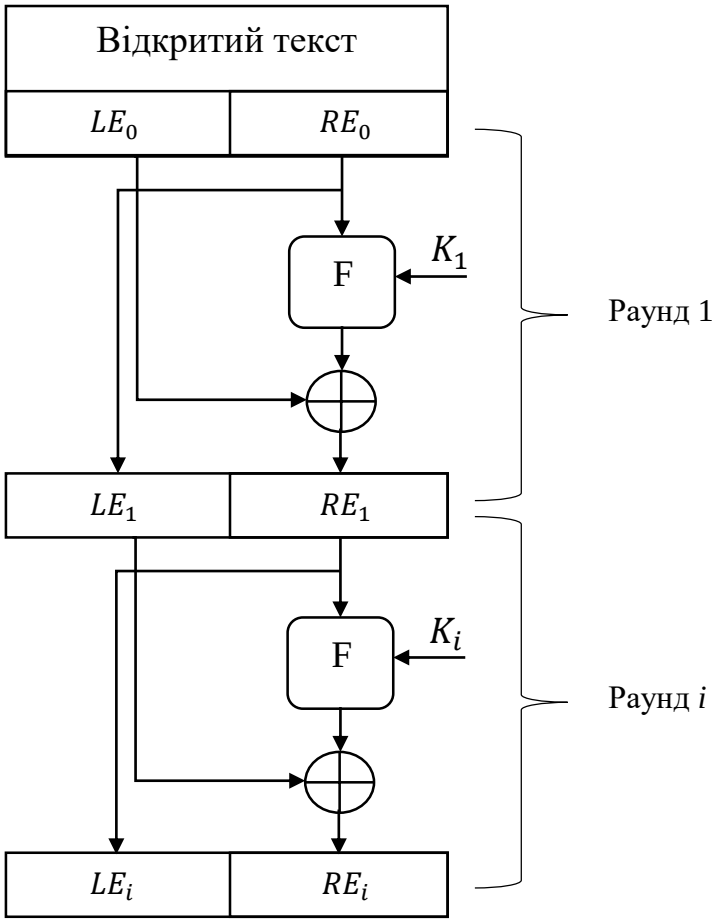


Рисунок 2.1 – Схема структури шифру Фейстеля

Звичайний текст ділиться на блоки фіксованого розміру, і за раз обробляється лише один блок. Отже, вхід до алгоритму шифрування — це блок звичайного тексту і ключ *K*.

Блок звичайного тексту розділений на дві рівні половини, які позначаються LE_0 як ліва половину блоку звичайного тексту і RE_0 – як права

половину блоку звичайного тексту. Тепер обидві ці половини блоку простого тексту (LE_0 і RE_0) проходять кілька раундів для створення блоку зашифрованого тексту.

У кожному раунді функція шифрування застосовується до правої половини LE_i блоку відкритого тексту разом з ключем N_i. Над результатом цієї функції шифрування потім виконується операція XOR з лівою половиною LE_i. Результатом функції XOR стає нова права половина для наступного раунду RE_{i+1}. Тоді як стара права половина RE_i стає новою лівою половиною LE_{i+1} для наступного раунду, як можна бачити на рисунку 2.2.

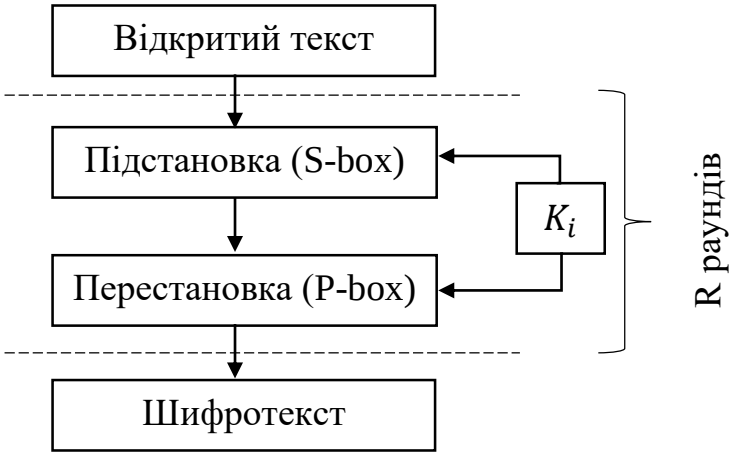


Рисунок 2.2 – Схема шифрування для SPN-мереж.

Виконанням раундової функції та операції XOR реалізується процес заміни, результат цих операцій передається в праву частину наступного раунду і попереднє значення правої частини передається як лівий операнд для наступного раунду, що забезпечує механізм перестановки. Ось як структура шифру Фейстеля представляє застосування заміни та перестановки по черзі, що схоже на структуру Шеннона.

Substitution–permutation network (SPN) – це серія пов’язаних математичних операцій, які використовуються в алгоритмах блокового шифрування таких як AES (Rijndael), 3-Way, Kalyna, Kuznyechik, PRESENT, SAFER, SHARK [25]. На відміну від мережі Фейстеля, SP-мережі обробляють за один раунд повністю блок даних.

Мережа такого типу використовує цілий блок шифротексту та ключа на вхід, після чого виконується декілька раундів або шарів блоків підстановки (S-блоків) та блоків перестановки (P-блоків) для отримання блоку зашифрованого тексту. Ключ для кожного раунду отримується у вигляді раундових ключів, отриманих від секретного ключа.

Розшифрування виконується таким самим чином тільки на основі обернених S- , P- блоків та раундових ключів у зворотному напрямку.

Багато сучасних блокових шифрів побудовані з використанням структури ARX — яка включає лише три операції: (A) модульне додавання, (R) обертання з фіксованими величинами та (X) XOR. Шифри, які побудовані на цій структурі – Speck, XXTEA [26]. Операції ARX популярні, тому що вони відносно швидкі та дешеві при реалізації в апаратному та програмному забезпеченні, їх виконання може бути надзвичайно простим, а також тому, що вони виконуються за постійний час, а отже, є несприйнятливими до атак із визначенням часу на виконання окремих криптографічних перетворень. Техніка раундового крипто аналізу, також використовується для того щоб зламувати раундові функції у шифрах, які використовують тільки операції ARX.

Причинами для використання SP-мережі в запропонованому методі є властивості перетворень, які використовуються в цих мережах. У процесі шифрування використовуються перетворення на основі S-боксів і P-боксів, які в свою чергу забезпечують властивості лавинного ефекту та є стійкими до криптоаналізу.

Отже, було проведено дослідження основних структур для побудови блокових шифрів. Наступним кроком є процес блокового шифрування для узагальнення існуючих підходів до шифрування, для якого доцільно виділити відмінності у запропонованому псевдонедетермінованому методі блокового шифрування.

2.3 Метод блокового шифрування

Проаналізувавши та дослідивши різні підходи до шифрування, структури, режими роботи блокових шифрів, задля покращення стійкості шифрування було запропоновано розробити метод псевдонедетермінованого блокового шифрування.

Як основу шифру обрано дві структури – SPN-мережа та схему Lai-Massey [27].

Схема Lai-Massey за конструкцією подібна до мережі Фейстеля з використанням раундової та напівраундової функції. Функція раунду — це функція, яка приймає два входи, підключ і блок даних, і повертає один вихідний блок рівний до довжини блоку даних. Напівраундова функція приймає два входи і перетворює їх на два виходи. Для будь-якого даного раунду вхідні дані розбиваються на дві половини, ліву і праву.

Спочатку вхідні дані передаються через функцію напівраунду. У кожному раунді різниця між входами передається функції раунду разом із підключом, а результат функції раунду потім додається до кожного входу. Потім вхідні дані передаються через функцію напівраунду. Потім це повторюється фіксовану кількість разів, і остаточним результатом є зашифровані дані. Завдяки своїй конструкції вона має перевагу над SPN мережею, оскільки раундову функцію не потрібно інвертувати - лише напівраундову, що дозволяє її легше інвертувати. Процеси шифрування та розшифрування досить подібні, розшифрування натомість вимагає зміни розкладу ключів, інвертованої функції напівраунду, а вихід функції раунду віднімається замість додавання.

Схема Lai-Massey є стійкою до атак на основі підібраного шифротексту та немає вразливостей до

Згідно до актуальних стандартів у блокових шифрах використовується ключ довжиною 128 біт, що задовольняє необхідний рівень стійкості. Також для побудови шифру використано режим шифрування CTR.

Отже, беручи до уваги розроблений математичний опис та обрані компоненти описані вище, опишемо алгоритм виконання процесу шифрування:

Крок 1. Користувач обирає файл, який необхідно зашифрувати або розшифрувати та вказує секретний ключ.

Крок 2. Файл даних розбивається на блоки по 32 біти, при необхідності виконується доповнення для кратності.

Крок 3. Секретний ключ розділяється на 4 частини (128 → 32 (4)), 32 біти будуть використані як вектор управління (v) і буде задавати режим роботи шифру на кожному раунді, решта – 3 блоки по 32 біти передаються у функції шифрування.

Крок 4. В залежності від значення вектору управління на кожному раунді використовується відповідна структура шифрування, якщо $v_i = 1$ (біт), тоді поточний блок даних буде зашифрований з використанням SPN мережі, якщо $v_i = 0$ – схеми Lai-Massey.

Крок 5. Виконується шифрування або розшифрування блоків даних, до тих пір поки не прочитано кінець файлу до кінця файлу.

Крок 6. Користувач отримує зашифрований або розшифрований файл.

Використання двох структур для виконання процесу шифрування додає елемент псевдонедетермінованості, натомість швидкість шифрування не змінюється. Такий підхід дозволяє підвищити стійкість до лінійного та диференційного криптоаналізу, оскільки при проведенні криптоаналізу зловмисник не зможе так швидко однозначно виділити алгоритм за яким виконується шифрування. Відповідно для успішного криптоаналізу необхідно буде витрати в два рази більше ресурсів та часу, оскільки використовується два різних підходи одночасно.

2.4 Процес розгортання раундових підключів

Для шифрування кожного блоку даних використовується набір з 32 раундових ключів. Питання забезпечення стійкості при генерації є критичним,

оскільки є безліч методів дослідження шифрів, які намагаються визначити слабкі раундові ключі (які в свою чергу мають менший обсяг чим повний секретний ключ), що є вразливістю алгоритму та може бути використано для його зламу.

При отриманні секретного ключа довжиною 128 біт, він ділиться на 4 частини по 32 біти – k_i, де $i = 0, ..., 3$. Далі перше значення частини ключа k_0 використовується для задання режиму роботи шифру на кожному з тридцяти двох раундів. k_1, k_2, k_3 відповідно беруть участь у генерації раундових ключів.

Для проектування стійкого генератора випадкових чисел є необхідність у повному періоді цього генератора. Тому було вирішено за основу генератора обрати значення поліномів, які будуть володіти властивістю незвідності. Згідно роботи У.Пітерсона, Е. Уелдона “Коди, які виправляють помилки” [28] в якій було досліджено поліноми щодо незвідності, не лінійності, корінь подвійного многочлена є незалежним, було обрано поліноми, які показали найкращі показники (класу F) – 0x00400007, 0x04C11DB7, 0x100D4E63 (у 16-ій системі числення).

Крок 1. Для кожного k_1, k_2, k_3 визначаємо значення старшого біта, для цього накладається відповідна маска.

Крок 2. Якщо старший біт дорівнює 0, тоді зсуваємо значення ключа на один біт ліворуч – $k_i \ll 1$.

Крок 3. Якщо старший біт дорівнює 1, тоді зсуваємо значення ключа на один біт ліворуч та виконується операція XOR між результатом зсуву та відповідним значенням поліному – $(k_i \ll 1) \oplus polinom_i$.

Крок 4. Виконується операція XOR над трьома поточними значеннями раундових ключів $k_1 \oplus k_2 \oplus k_3$. Таким чином отримано раундовий ключ довжиною 32 біти.

З експертної точки зору такий підхід для отримання раундового ключу шифрування повинен давати стійкі значення. Для перевірки даного твердження далі необхідно буде провести низку тестувань.

2.5 Висновки до розділу

У даному розділі було досліджено відомі структури процесу блокового шифрування, на основі яких будуються сучасні алгоритми шифрування.

Виконано математичний опис – узагальненого процесу блокового шифрування та розробленого методу псевдонедетермінованого шифрування, який дозволяє уніфікувати запропонований підхід до шифрування.

Описано метод блокового псевдонедетермінованого шифрування файлів. Для реалізації даного методу необхідно розробити алгоритми та виконати реалізації засобу шифрування відповідно до запропонованого методу.

3 РОЗРОБКА ЗАСОБУ ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ

3.1 Обґрунтування вибору засобів для розробки

Для реалізації засобу псевдонедетермінованого блокового шифрування файлів є низка вимог, які потрібно враховувати при виборі мови програмування, програмних засобів та підходів у програмуванні:

- готовий додаток повинен працювати на будь-якій операційній системі (ОС);
- мова програмування повинна бути безпечною та захищеною;
- виконувати необхідні криптографічні перетворення за постійний час;
- мати можливість розширення функціоналу.

Описані вище вимоги направлені на вибір оптимальних рішень, які допоможуть якісно реалізувати запропонований засіб шифрування, базуючись на сучасних функціональних та економічних вимогах.

Для вибору мови програмування було проаналізовано декілька найбільш популярних, серед них Python [29], C, C++, C# [30], Java. За весь час існування мова програмування Java [31] була найбільш стабільною та завжди показувала свою ефективність при використанні у різних сферах, особливо для розробки серверною частини різноманітних систем. Також дана мова має велику аудиторію користувачів, що дозволяє з легкістю вирішувати проблемні питання, а також постійно розвивається відповідно до сучасних трендів, активно підтримується розробниками. Виходячи з поставленою задачі, щодо написання засобу для шифрування файлів та оцінивши переваги мови програмування Java, було обрано саме її.

З огляду на вказані вище критерії було обрано мову програмування Java. Оскільки, першочерговою задачею даної мови є забезпечення незалежності від ОС. Використовуючи Java немає необхідності написання окремого коду для підлаштування під інші операційні системи та процесори. Тому, що віртуальна машина Java інтерпретує написаний код в зрозумілий для поточної ОС.

Java є безпечною мовою програмування, оскільки має строгу типізацію

даних, доступ до даних задається з використанням модифікаторів доступу, присутнє автоматичне управління оперативною пам’яттю (з можливістю додаткового налаштування), запуск у вигляді байт-коду, який передбачає попередню компіляцію, що попереджає виникненню помилок.

Java є досить зручною мовою для написання шифрів різного роду, оскільки розроблені функції можуть бути використані у інших проектах та системах, як зовнішні бібліотеки.

Також сучасний стан мови дозволяє реалізувати інтерфейс засобу, який буде використовуватись як веб-додаток. Що спростить використання методу користувачам.

Написання проекту відбувалось у середовищі IntelliJ IDEA (яке є найбільш прогресивним варіантом серед інших аналогів та має низку переваг), з підтримкою Maven – фреймворк, який дозволяє автоматизувати етапи збірки проекту, фреймворку Spring [32] – задає каркас розробленого засобу, фреймворку Vaadin – який дозволяє з легкістю реалізовувати сучасні інтерфейси безпосередньо на мові Java.

Для того, щоб спостерігати за станом виконання програми було запропоновано додати логування подій. За даним процесом закріплена бібліотека Log4J, яка дозволяє нотувати повідомлення різного класу.

Для проведення тестування блоків програми використано стандарт Special Publication 800-22 Revision 1a, згідно до NIST [33], який направлений на визначення ефективності генерації випадкових значень. Що є доцільним для блокового шифрування, яке має на меті змінити вхідні дані на такі, що буде важко відновити початковий стан.

3.2 Узагальнена архітектура засобу шифрування

Розроблений засіб буде реалізовано у вигляді бібліотеки, яку можливо буде використовувати у будь-яких системах. Основні функції програми це виконання шифрування та розшифрування файлів. Для зручної демонстрації було розроблено інтерфейс, який дозволяє графічно обирати файли,

виконувати операції шифрування та розшифрування, завантажувати результат роботи засобу. На рисунку 3.1 зображена структура роботи засобу шифрування файлів.

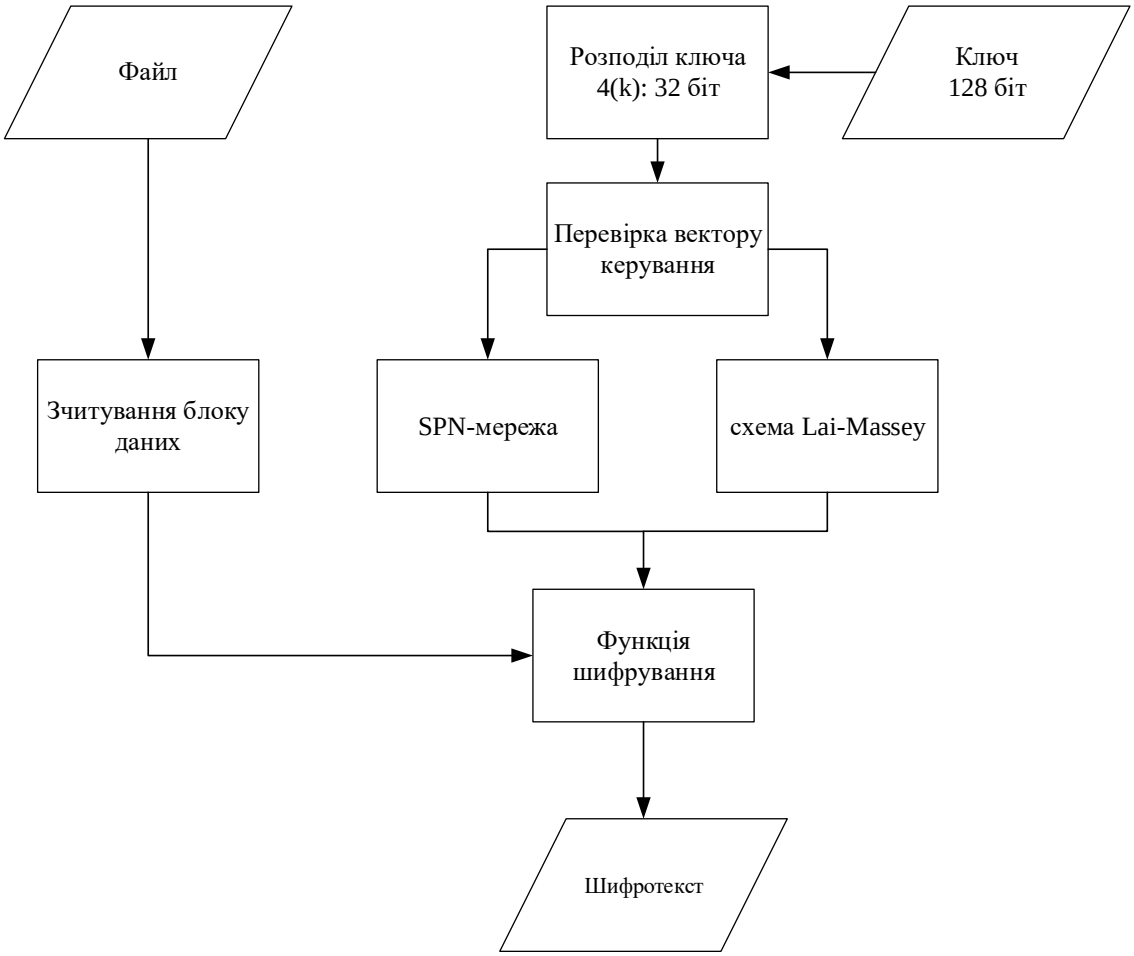


Рисунок 3.1 – Структурна схема псевдонедетермінованого блокового шифрування файлів

Інтерфейс користувача містить два компоненти – вибір файлу для шифрування/розшифрування та вибір секретного ключа. Процес отримання ключа додано для емуляції справжнього процесу шифрування і є нерелевантним рішенням для практичного використання. При подальшому використанні засобу є необхідність більш детального налаштування методу отримання секретного ключа, який в свою чергу повинен вже знаходитись в системі в якій буде виконуватись програмний модуль.

Блок вибору файлу зчитує файл будь-якої довжини в буфер засобу під час його виконання та видаляється після закінчення процесів чи то шифрування, чи то розшифрування.

Блок розгортання вектору ініціалізації, який відповідає за управління режимом шифрування на кожному раунді та блок розгортання раундових підключів, на основі яких отримуються раундові ключі, для виконання шифрування.

Найголовніший блок – це блок криптографічних перетворень, який є уособленням саме процесу шифрування та розшифрування в залежності значення вектору ініціалізації.

Узагальнена архітектура засобу псевдонедетермінованого блокового шифрування файлів зображена на рисунку 3.2.

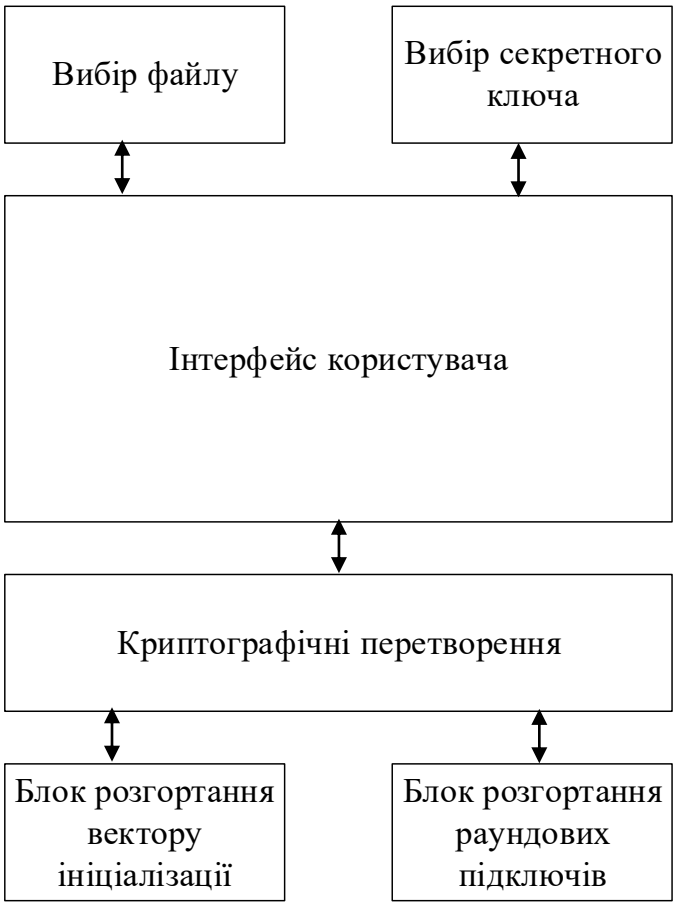


Рисунок 3.2 – Узагальнена архітектура засобу псевдонедетермінованого блокового шифрування

Після розробки архітектури необхідно перейти до опису основних блоків засобу блокового шифрування відповідно до запропонованого архітектурного рішення.

3.3 Розробка інтерфейсу засобу шифрування

Для засобу псевдонедетермінованого блокового шифрування файлів було запропоновано розробити інтерфейс, для зручної взаємодії для користувачів. Інтерфейс реалізований з використанням бібліотеки (фреймворку) Vaadin, яка дозволяє засобами Java писати компоненти інтерфейсу так, якби це були JavaScript та CSS.

Vaadin Framework - це UI фреймворк з серверною моделлю програмування, в якій вся логіка UI та його стан розташовані на сервері, а в браузері виконується лише код UI компонентів. По суті, це технологія тонкого клієнта, де браузер лише відображає те, що скаже сервер, а всі події відправляються на сервер.

Серверний підхід дозволяє забути про те, що технологія ведеться під веб, і розробляти UI як настільний Java додаток з безпосереднім доступом до даних та сервісів на сервері. При цьому Vaadin подбає і про відображення UI в браузері, і про AJAX-взаємодію між браузером та сервером. Двигун Vaadin здійснює рендеринг інтерфейсу програми серверної сторони в браузері і реалізує всі деталі обміну клієнта і сервера.

Для взаємодії з основними компонентами застосунку було розроблено інтерфейс, який зображено на рисунку 3.3.

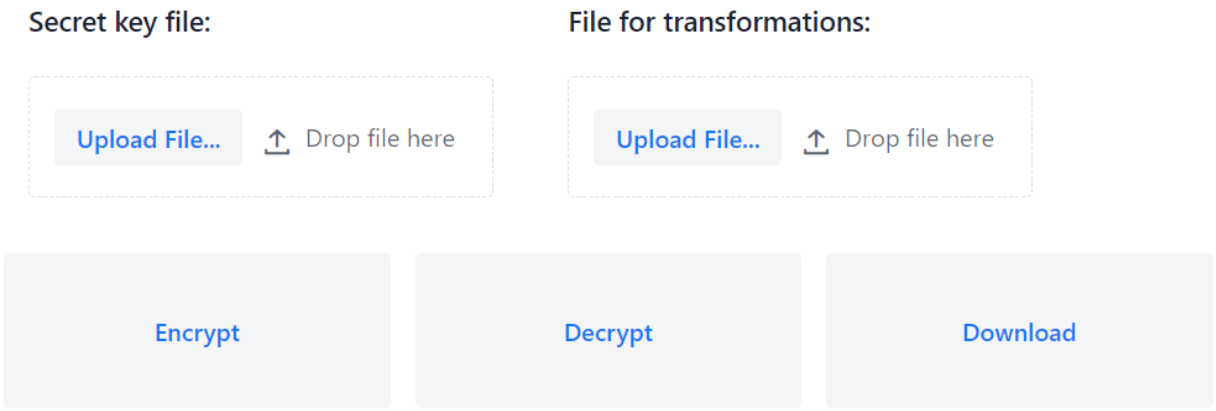


Рисунок 3.2 – Інтерфейс засобу шифрування

Інтерфейс містить п’ять компонентів. Два компоненти для завантаження секретного ключа та самого файлу, який необхідно зашифрувати.

Інші компоненти – кнопки, для вибору таких опцій:

1) Encrypt – виконує процес шифрування вказаного файлу;
2) Decrypt – виконує процес розшифрування відповідного файлу;
3) Download – кнопка призначена для завантаження файлу зашифрованого або розшифрованого, в залежності від обраної раніше опції.

Функціональна частина інтерфейсу описана у класі MainView.

Далі доцільно перейти до розробки блоків, які відповідають за криптографічні перетворення.

3.4 Розробка блоків розгортання вектору управління та раундових ключів

Для реалізації блоків розгортання вектору управління та раундових ключів було визначено окремий клас – KeyTransformations.

Після того як користувач обрав файли з секретним ключем та файлом, який необхідно зашифрувати виконується генерація раундових ключів.

На рисунку 3.3 зображено алгоритм розгортання раундових ключів.

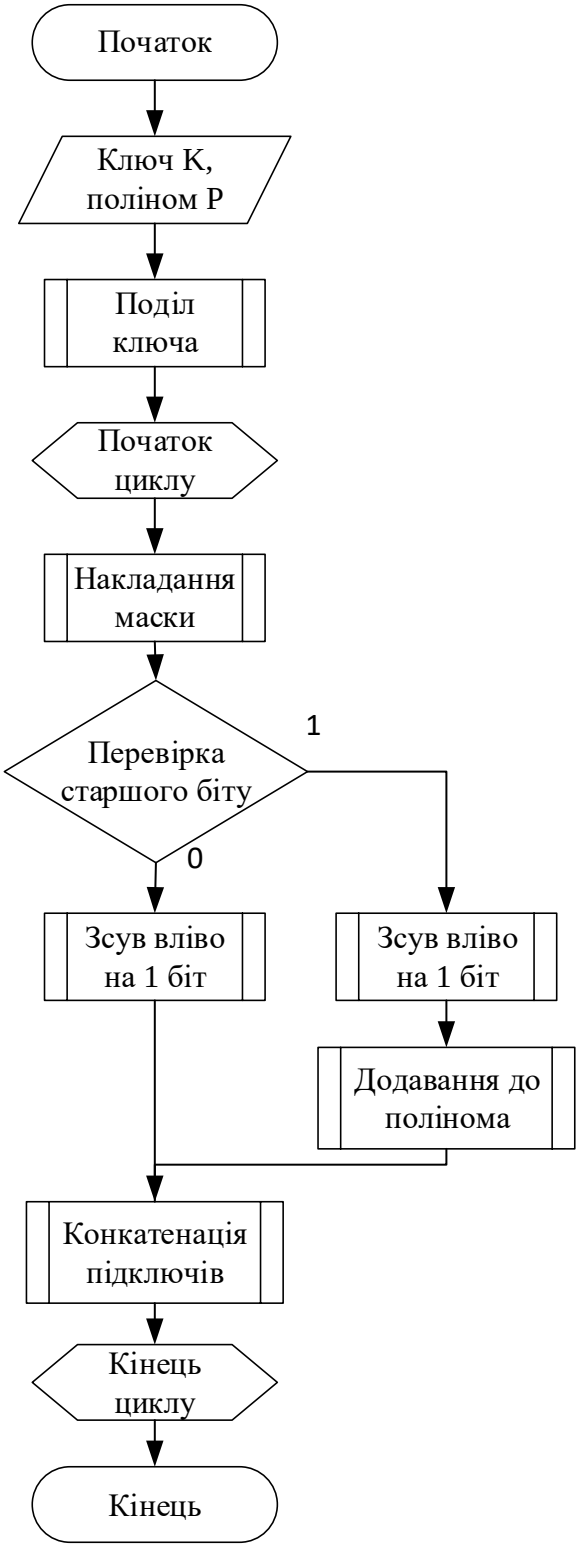


Рисунок 3.3 – Алгоритм роботи блоку розгортання раундових ключів

У класі KeyTransformations вбудовані статичні значення поліномів, які є основою генератора ключів.

Спочатку ключ ділиться на 4 частини, відповідно одна з них буде складати вектор управління *V*, інші 3 будуть використовуватись далі для

розгортання раундових ключів (алгоритм поділу секретного ключа розглянуто на рисунку 3.4).

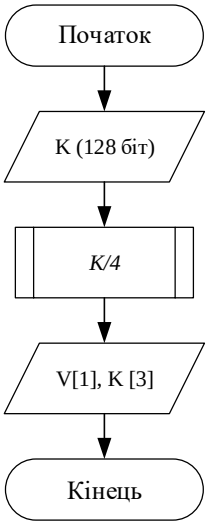


Рисунок 3.4 – Алгоритм процедури розбиття секретного ключа

Далі у циклі на тридцять дві ітерації виконуються такі кроки (оскільки наступні процедури є досить важливими при реалізації даного методу шифрування, буде доцільно зобразити :

Крок 1. Накладання маски на кожен з підключів K_i, алгоритм виконання даної процедури зображено на рисунку 3.5.

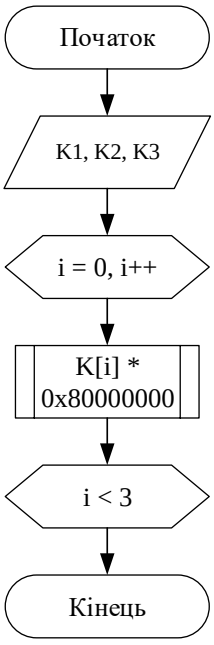


Рисунок 3.5 – Алгоритм процедури накладання маски на кожен підключ, для визначення старшого біта

Крок 2. Перевірка значення старшого біта кожного підключа, що зображена на рисунку 3.6;

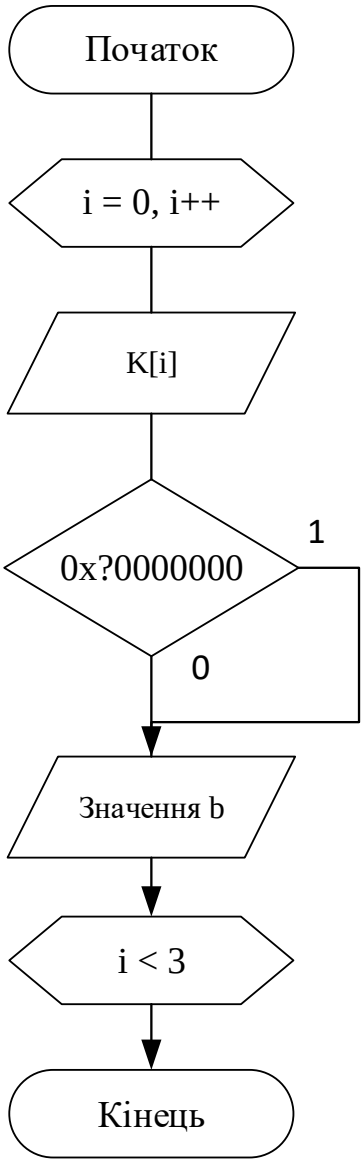


Рисунок 3.6 – Алгоритм процедури перевірки старшого біту для кожного підключа

Крок 3. Залежно від значення старшого біта виконуються базові криптографічні перетворення – циклічний зсув ліворуч на 1 біт та операція XOR, що зображено на рисунку 3.7.

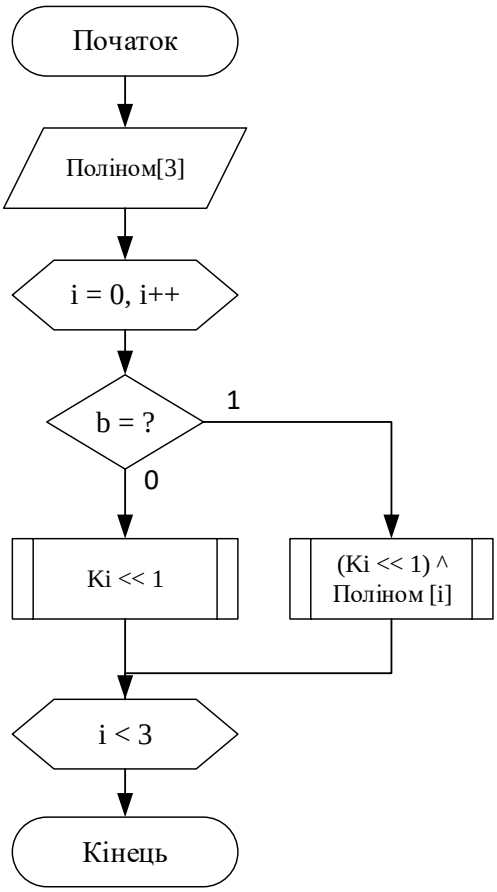


Рисунок 3.7 – Алгоритм процедури перетворень підключів

Крок 4. Результат обчислень кожного з підключів використовується далі для формування раундового ключа за допомогою операції XOR між підключами, що зображено на рисунку 3.8.

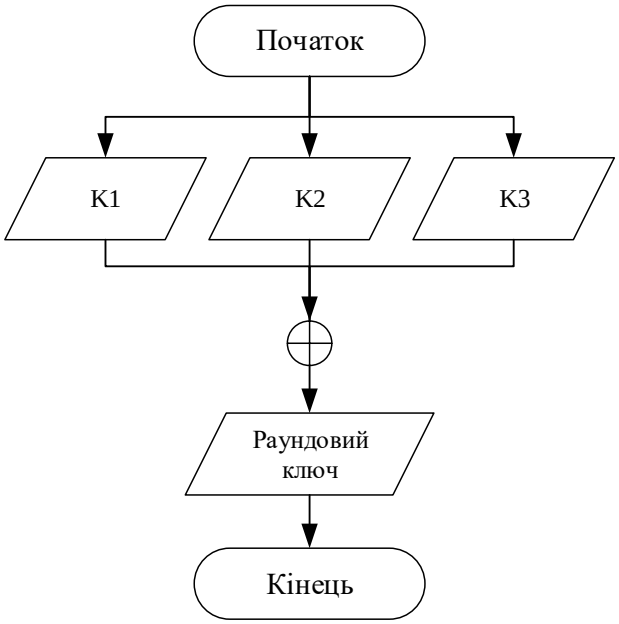


Рисунок 3.8 – Алгоритм процедури формування раундового ключа на основі попередніх результатів обчислень

Кожне значення раундового ключа передається у блок криптографічних перетворень для подальшого шифрування блоку даних. Перейдемо до розробки блоку криптографічних перетворень.

3.5 Розробка блоку криптографічних перетворень

Відповідно до узагальненої архітектури засобу (рис. 3.1) є необхідність реалізувати блок криптографічних перетворень як окрему частину у програмному засобі. Для цього було виділено окремий клас у застосунку – CryptoTransformations. Алгоритм роботи блоку криптографічних перетворень зображено на рисунку 3.4.

Отже, на кожному з тридцяти двох раундів шифрування для кожного блоку визначається алгоритм шифрування в залежності від значення вектору керування. Після визначення режиму шифрування, у клас CryptoTransformations передається наступне значення раундового ключа та виконуються обчислення вихідного блоку даних.

Якщо вектор керування дорівнює 1, виконується шифрування відповідно до структури SP–мережі, тобто послідовно обчислюються S- та P-бокси. У випадку якщо вектор керування рівний 0 шифрування відбувається згідно до схеми Lai-Massey:

Крок 1. Вхідний блок даних розбивається навпіл;

Крок 2. Над отриманими половинами блоків виконується неповна функція *H*;

Крок 3. Обчислюється значення побітового віднімання результатів виконання функції *H*;

Крок 4. Обчислюється функція шифрування *F*, з використанням раундового ключа;

Крок 5. Над результатом обчислення функції *F* та двома результатами функцій *H* виконується операція XOR.

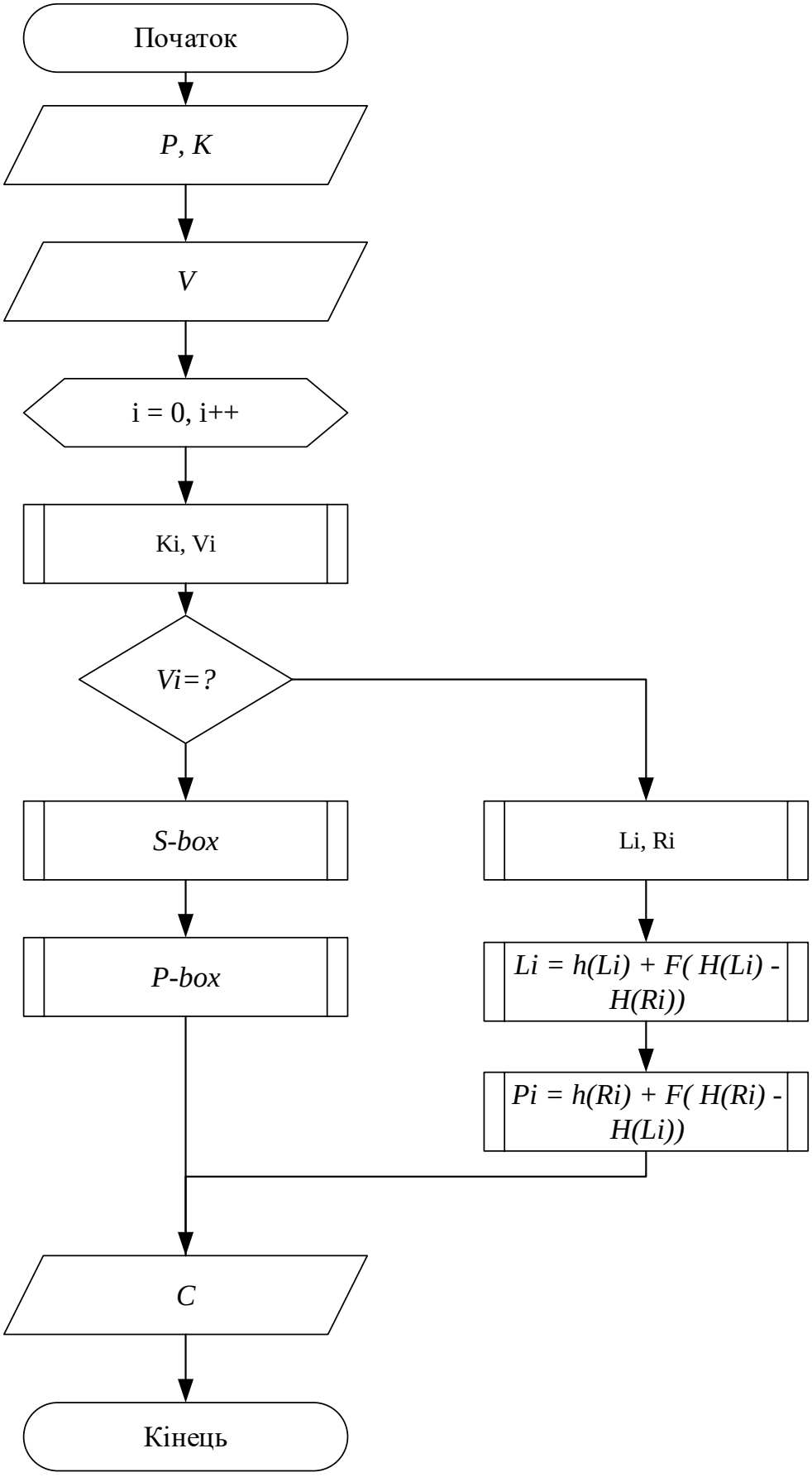


Рисунок 3.4 – Алгоритм роботи засобу шифрування файлів (блок криптографічних перетворень)

На виході обох структур шифрування отримуємо шифротекст. Далі отримані шифротексти конкатенуються та повертаються користувачеві у форматі файлу.

3.6 Висновки до розділу

Отже, у даному розділі було представлено узагальнену архітектуру засобу шифрування файлів, який складається з двох основних блоків – інтерфейс та блок криптографічних перетворень.

Відповідно до архітектури засобу було представлено алгоритми реалізації блоку криптографічних перетворень та алгоритм роботи засобу в цілому. Обґрунтовано вибір програмних та семантичних засобів, як то мова програмування, середовище розробки, необхідні компоненти, які будуть використані для розробки програмного модуля.

Базуючись на розроблених алгоритмах можемо переходити до реалізації засобу та проведення апробації результатів, шляхом написання автоматизованих тестів відповідно до сучасних стандартів.

4 ТЕСТУВАННЯ ЗАСОБУ ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ

4.1 Обґрунтування вибору методики тестування

Тестування програмного забезпечення [34] — це метод перевірки, чи відповідає фактичний програмний продукт очікуваним вимогам, і переконатися, що програмний продукт не містить дефектів. Він включає виконання програмних/системних компонентів за допомогою ручних або автоматизованих інструментів для оцінки однієї або кількох властивостей, що цікавлять. Метою тестування програмного забезпечення є виявлення помилок, прогалин або відсутніх вимог на відміну від реальних вимог.

Тестування програмного забезпечення є важливим, оскільки якщо в програмному забезпеченні є якісь баги або помилки, їх можна виявити на ранніх стадіях і вирішити до релізу програмного продукту для клієнтів/користувачів. Правильно перевірений програмний продукт забезпечує надійність, безпеку та високу продуктивність, що в подальшому призводить до економії часу, економічної ефективності та задоволеності клієнтів.

Тестування важливе, оскільки помилки програмного забезпечення можуть бути дорогими або навіть небезпечними. Помилки програмного забезпечення потенційно можуть призвести до грошових і людських втрат, і історія повна таких прикладів.

Переваги використання тестування програмного забезпечення:

1) Економічна ефективність – це одна з важливих переваг тестування програмного забезпечення. Своєчасне тестування будь-якого IT-проекту допоможе заощадити гроші в довгостроковій перспективі. Якщо помилки виявлені на ранньому етапі тестування програмного забезпечення, їх виправлення коштує дешевше.

2) Безпека – це найбільш вразлива і чутлива перевага тестування програмного забезпечення. Люди шукають надійні продукти. Це допомагає усунути ризики та проблеми раніше.

3) Якість продукту – це важлива вимога будь-якого програмного продукту. Тестування гарантує, що тільки якісний продукт буде доставлений клієнтам.

4) Задоволеність клієнта – головна мета будь-якого продукту - задовольнити своїх клієнтів. Тестування UI/UX забезпечує найкращий досвід користувача.

Відповідно до ANSI/IEEE 1059 [35], тестування в інженерії програмного забезпечення — це процес оцінки програмного продукту, щоб визначити, чи відповідає поточний програмний продукт необхідним умовам чи ні. Процес тестування включає оцінку функцій програмного продукту на вимоги з точки зору будь-яких відсутніх вимог, помилок або помилок, безпеки, надійності та продуктивності.

Зазвичай тестування поділяють на три категорії:

– Функціональне тестування;
– Нефункціональне тестування або тестування продуктивності;
– Технічне обслуговування (регресія та обслуговування);

Для проведення тестування розробленого засобу псевдонедетермінованого блокового шифрування було обрано функціональний тип тестування, а саме модульні тести.

Модульні тести [36] – це тип тестування програмного забезпечення, при якому перевіряються окремі блоки або компоненти програмного забезпечення. Мета полягає в тому, щоб підтвердити, що кожна одиниця програмного коду працює належним чином. Модульне тестування проводиться під час розробки програми розробниками. Модульні тести виділяють блок коду та перевіряють його правильність. Блоком може бути окрема функція, метод, процедура, модуль або об'єкт.

Щоб провести модульне тестування, розробники пишуть розділ коду для перевірки певної функції в програмному застосунку. Розробники також можуть ізолювати цю функцію для більш ретельного тестування, що виявляє непотрібні залежності між функцією, що тестується, та іншими блоками, щоб можна було усунути залежності. Розробники зазвичай використовують фреймворк UnitTest [37], який зазвичай має реалізацію на більшості сучасних мовах програмування, для розробки автоматизованих тестових випадків для модульного тестування.

Оскільки засіб для шифрування був розроблений на мові програмування Java є доцільним використати фреймворк JUnit [38]. JUnit є одним з найпопулярніших фреймворків модульного тестування в екосистемі Java.

Із застосуванням даних інструментів доцільно перейти до тестування окремих компонентів засобу відповідно до методики модульного тестування. Базуючись на інтерфейсі засобу, можна виділити 5 базових компонентів програмного додатку – метод отримання секретного ключа, метод отримання файлу для перетворень, метод шифрування, метод розшифрування та метод завантаження файлу з результатами.

4.2 Тестування стійкості шифрування

Метод отримання секретного ключа. У даній реалізації процес отримання секретного ключа направлений на зчитування інформації з файлу, який було вказано користувачем.

Відповідно до реалізації методу шифрування була поставлена вимога, щоб довжина ключа дорівнювала 128 біт. Тому необхідно виконати перевірку за допомогою написання тест-кейсу, який буде перевіряти довжину отриманого ключа (результат виконання тест-кейсу зображений на рисунку 4.1).

```java
@SpringBootTest
class BlockCipherApplicationTests {

    @Test
    void keyLengthTest() {
        KeyTransformations key = new KeyTransformations();
        String keyValue = key.getSecretKey();
        StringBuilder result = new StringBuilder();
        char[] chars = keyValue.toCharArray();
        for (char aChar : chars) {
            result.append(
                    String.format("%8s", Integer.toBinaryString(aChar))
                            .replaceAll( regex: " ", replacement: "0")
            );
        }
        assertEquals( expected: 128,result.length());
    }
```

Рисунок 4.1 – Код, який реалізує тест-кейс перевірки ключа

Також додатково було описано тест-кейс направлений на перевірку чи файл, який містить секретний ключ не є порожнім.

На рисунку 4.2 можемо бачити результат виконання, який показує що тести було пройдено успішно, отже застосунок реалізовано вірно.

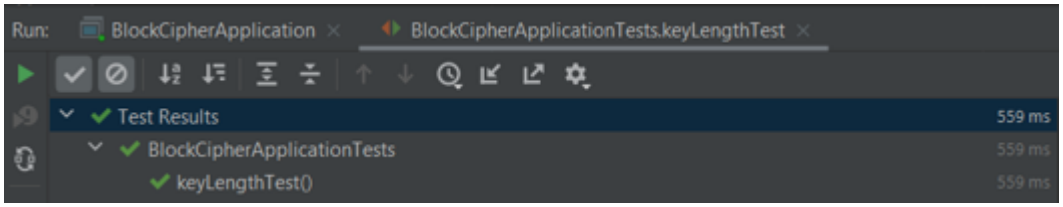


Рисунок 4.2 – Тест-кейс для перевірки процесу отримання ключа

Для функціонального тестування інших компонентів буде достатньо перевірити їх на отримання результату виконання методів, які описують дії цих компонентів. Що буде свідчити що отриманий результат збігається з вимогами щодо типів даних та перевірки на наявність таких результатів в загальному. Для цього було написано низку тест-кейсів, які показали успішний результат проходження.

4.3 Проведення тестування алгоритму шифрування

Для оцінки результатів роботи самого методу шифрування не доцільно використовувати типові методи тестування. Тому було визначено, що

необхідно реалізувати набір статистичних тестів відповідно до стандартів NIST SP 800-22 (Набір статистичних тестів для випадкових і псевдовипадкових генераторів чисел для криптографічні засобів). Даний стандарт тестування доцільно використовувати через те, що процедура шифрування має на меті з використанням вхідного відомого значення перетворити у псевдовипадкову послідовність.

В ідеальному варіанті значення отриманого шифротексту повинно бути згенеровано випадкову послідовність у співвідношенні, що кожний біт інформації буде змінений з імовірністю 50%.

Випадкова послідовність може бути охарактеризована і описана в термінах ймовірності. Імовірний результат статистичних тестів, якщо його застосувати до дійсно випадкової послідовності, відомий апріорі і може бути описаний у імовірнісному підході. Існує нескінченна кількість можливих статистичних тестів, кожен з яких оцінює наявність або відсутність «шаблону», який, якщо буде виявлений, вказує на не випадковість послідовності. Оскільки існує дуже багато тестів для оцінки того, чи є послідовність випадковою чи ні, жоден конкретний кінцевий набір тестів не вважається «повним». Крім того, результати статистичного тестування слід інтерпретувати з певною обережністю, щоб уникнути неправильних висновків щодо конкретної реалізації алгоритму.

Отже, серед різноманіття тестів, які описано у даному стандарті, було обрано 3 основних тести перевірки випадкових значень – частотний тест (для послідовності біт), частотний тест для блоків даних, Runs тест.

Далі слід окремо розглянути процес виконання кожного з тест-кейсів.

Частотний тест. Даний тест має на меті визначити співвідношення нулів та одиниць для заданої послідовності біт. Очікуваний результат, що частка нулів та одиниць повинна бути приблизно рівна для дійсно випадкової послідовності. Тест оцінює близькість частки одиниць до ½ , тобто кількість одиниць та нулів у послідовності має бути приблизно однаковою. Усі

подальші тести залежать від результату проходження поточного тесту. Доцільно буде перейти до опису тесту:

1) Перетворення на ±1. Нуля та одиниці з отриманої послідовності (ε) перетворюються на значення –1 та +1 відповідно. Далі ці значення додаються для отримання $S_n = X_1 + X_2 + \cdots + X_n$, де $X_i = 2\varepsilon_i - 1$.

2) Обчислюється значення статистики тесту $S_{obs} = \frac{|S_n|}{\sqrt{n}}$.

3) Обчислюється значення $P = erfc\left(\frac{S_{obs}}{\sqrt{2}}\right)$, де $erfc$ це функція додаткової помилки.

Правило перевірки отриманого значення P говорить про те, що якщо $P < 0{,}01$, тоді ця послідовність не є випадковою. В іншому випадку послідовність можливо вважати випадковою.

Результати виконання даного тест-кейсу показали, що значення P = 0,0592 (результат зображено на рисунку 4.3).

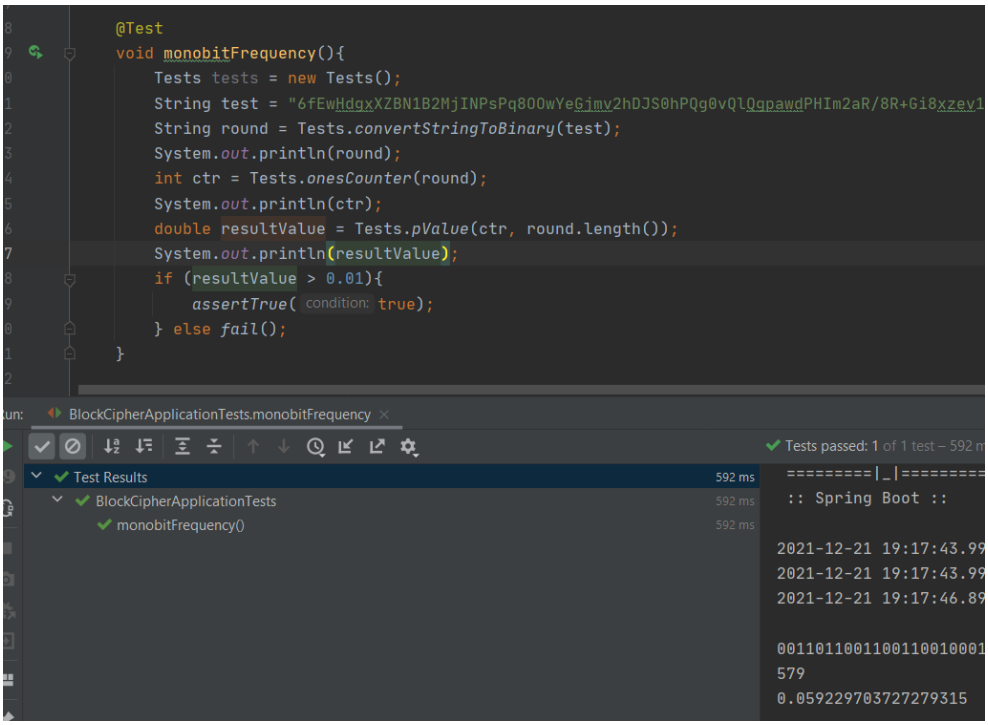


Рисунок 4.3 – Результат виконання частотного тесту послідовності

Оскільки було успішно пройдено частотний тест на послідовності шифротексту, доцільно буде перейти до виконання наступного тест-кейсу – частотний тест (для блоків).

Метою виконання даного тесту є визначення частки одиниць для *M*-бітових блоків шифротексту. А саме чи є частота одиниць у кожному блоці рівною приблизно *M/2*, як можна було б очікувати від випадкової послідовності. Якщо розмір блоку дорівнює 1, тест можна спростити то звичайного частотного тестування, яке було розглянуто раніше.

Постановка даного тесту має такий вигляд:

1) Розподілення послідовності на блоки що не перетинаються

$$N = \left\lfloor \frac{n}{M} \right\rfloor,$$

де n – довжина повідомлення у бітах;

M – довжина кожного блоку.

1) Визначається значення пропорції π_i одиниць до кожного *M*-бітного блоку використовуючи формулу:

$$\pi_i = \frac{\sum_{j=1}^{M} \varepsilon_{(i=1)M+j}}{M},$$

для $1 \leq i \leq N$.

2) Обчислюється статистика X^2 (Міра того, наскільки добре спостерігається частка одиниць у даному М-бітному блоці та відповідає очікуваній пропорції (1/2):

$$X^2(obc) = 4M \sum_{i=1}^{N} (\pi_i - 1/2)^2.$$

3) Обчислюється значення $P = igame(\frac{N}{2}, \frac{X^2(obc)}{2})$, де $igame$ є неповною гамма-функцією для Q(a,x)

Для даного тесту правило перевірки є ідентичним до попереднього випадку частотного тестування, тобто очікується, що $P < 0{,}01$. Програмна реалізація вище описаного тесту наведена на рисунку 4.4.

```
public static double pbValue(final int ones, final int n) {
    final int blockSize = (int) Math.sqrt(n);
    System.out.println("blockSize = " + blockSize);
    final int blockCount = n / blockSize;
    System.out.println("blockCount = " + blockCount);
    Fraction sum = Fraction.ZERO;
    for (int i = 0; i < blockCount; i++) {
        final Fraction p = new Fraction(ones, blockSize);
        final Fraction f = p.subtract(Fraction.ONE_HALF);
        sum = f.multiply(f).add(sum);
    }
    System.out.println("sum = " +sum);
    final double xSquared = new Fraction( num: 4).multiply(
            new Fraction(blockSize)).multiply(sum).doubleValue();
    System.out.println("xSquared = " + xSquared);
    igamc( a: blockCount / 2.0,  x: xSquared / 2.0);
```

Рисунок 4.4 –Програмна реалізація частотного тест-кейсу для блоків даних

Результати виконання даного тест-кейсу показали, що значення P = 0,706 (результат зображено на рисунку 4.5).

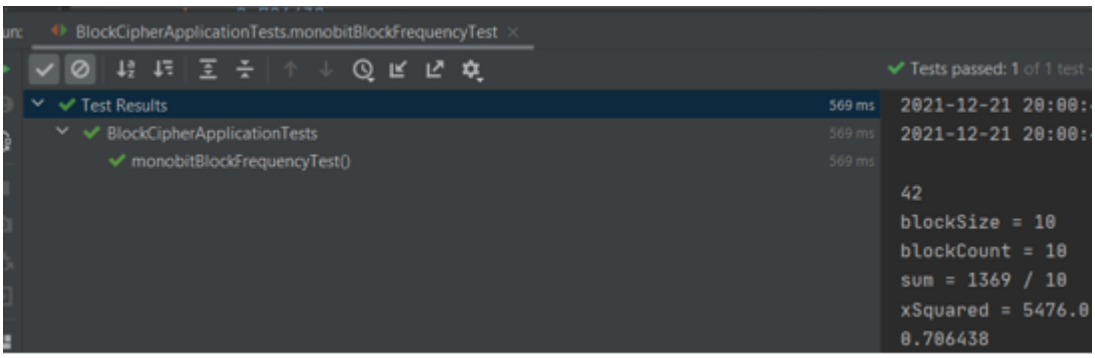


Рисунок 4.5 – Результат виконання частотного тесту послідовності блоків

Частотний тест послідовності блоків було також успішно пройдено, оскільки 0,706 > 0,01, тому перетворення можна вважати випадковими.

Наступний тест, це Runs тест. Тест такого типу направлений на дослідження загальної кількості запусків для послідовності, де запуск – це безперервна послідовність ідентичних бітів. Пробіг довжини k складається точно з k – ідентичних бітів і обмежений до та після цієї послідовності бітом протилежного значення. Мета тесту – визначити, чи відповідає кількість циклів одиниць та нулів різної довжини очікуваним для випадкової послідовності. Зокрема, цей тест визначає, чи є коливання між нулями та одиницями занадто швидкими чи повільними.

Опис процесу виконання даного тесту наведено далі:

1) Обчислюється передтестова послідовність π одиниць у вхідній послідовності : $\pi = \frac{\sum_j \varepsilon_j}{n}$.
2) Визначається, чи було пройдено тест на частоту.

   Оскільки спостережуване значення π, яке було визначено раніше, знаходиться у вибраних межах, можна застосувати даний тест.
3) Обчислюється статистика тесту $V_n(obs)$(Загальна кількість запусків (тобто загальна кількість нульових запусків + загальна кількість одиничних запусків) у всіх n бітах.): $\sum_{k=1}^{n-1} r(k) + 1$, де $r(k) = 0$, якщо $\varepsilon_k = \varepsilon_{k+1}$, та $r(k) = 1$ навпаки.
4) Обчислюється значення $P = erfc\left(\frac{|V_n(obs) - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right)$.

Якщо обчислене значення $P < 0{,}01$, то робимо висновок, що послідовність невипадкова. Інакше робимо висновок що послідовність випадкова.

Результати виконання даного тест-кейсу показали, що значення P = 0,706438 (результат зображено на рисунку 4.6).

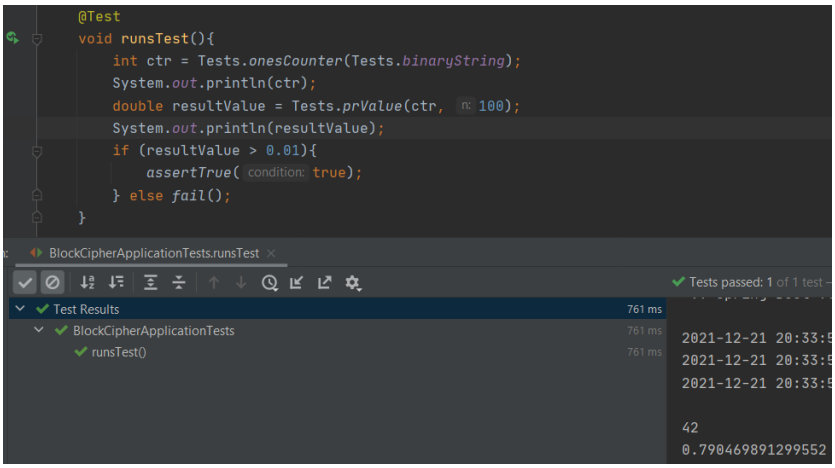


Рисунок 4.6 – Результат виконання Runs тесту

За результатами даного тесту отримана послідовність шифротексту також вказує її відповідність до вимог щодо випадковості.

4.4 Висновки до розділу

У даному розділі було розглянуто підходи до тестування програмних засобів. Визначено, які підходи слід розглянути для засобу псевдонедетермінованого блокового шифрування файлів.

Проведено функціональне тестування компонентів програмного засобу, яке показало працездатність даних компонентів. Було визначено, що засіб працює коректно

Відповідно до стандартів тестування засобів для отримання випадкових та псевдовипадкових послідовностей було реалізовано та пройдено низку тест-кейсів. Результати виконання набору тестів показало, що розроблений алгоритм шифрування може бути практично використаний відповідно до вимог тестування.

5 ЕКОНОМІЧНА ЧАСТИНА

Метою економічної частини магістерської кваліфікаційної роботи є обґрунтування економічної доцільності розробки методу та засобу псевдонедетермінованого блокового шифрування. Для цього необхідно виконати такі етапи робіт:

- оцінити комерційний потенціал розробки;
- спрогнозувати витрати на виконання наукової роботи та впровадження її результатів;
- спрогнозувати комерційний ефект від реалізації результатів розробки;

5.1 Оцінювання комерційного потенціалу розробки (технологічний аудит розробки)

Об’єктом дослідження магістерської кваліфікаційної роботи є процес забезпечення надійного шифрування файлів.

Було залучено трьох незалежних експертів для проведення технологічного аудиту, а саме: Баришев Ю. В., Куперштейн, Л. М., Лужецький В.А. Кожному експерту була надана таблиця, яку він повинен заповнити своїми оцінками після ознайомлення із запропонованою розробкою. Дана таблиця визначатиме критерії оцінювання потенціалу розробки та їх можливу оцінку в балах. Після виконання цього, буде обрахована середньоарифметична сума балів та буде визначено рівень економічного потенціалу розробки. Критерії оцінювання потенціалу розробки наведено у додатку Б.

Таблиця 5.1 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали експерта		
	Баришев Ю. В.	Куперштейн Л. М.	Лужецький В. А.
	Бали, виставлені експертами:		
1	3	3	4
2	2	4	2
3	4	3	3
4	3	4	4
5	4	3	4

Продовження таблиці 5.1

Критерії	Прізвище, ініціали експерта		
	Баришев Ю. В.	Куперштейн Л. М.	Лужецький В. А.
	Бали, виставлені експертами:		
6	3	2	4
7	3	3	3
8	2	4	2
9	4	3	4
10	4	4	3
11	3	4	4
12	3	3	3
Сума балів	$СБ_1$=38	$СБ_2$=40	$СБ_3$=40
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^i СБ_i}{i} = \frac{38+40+40}{3} = 39{,}3$		

За отриманими результатами оцінки із таблиці 5.1, можна зробити висновок базуючись на таблиці 5.2 рівнів комерційного потенціалу розробки.

Таблиця 5.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів	Рівень комерційного потенціалу
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-50	Високий

Відповідно до таблиці 5.2, рівень комерційного потенціалу розробки методу блокового шифрування є вище середнього.

Оцінювання рівня якості розробки методу блокового шифрування здійснюється з метою порівняльного аналізу та визначення найбільш ефективного, з технічної точки зору, варіанта інженерного рішення.

Рівень якості – кількісна характеристика міри придатності певного виду продукції для задоволення конкретного попиту на неї при порівнянні з відповідними базовими показниками за фіксованих умов споживання.

Абсолютний рівень якості розробки методу блокового шифрування потрібно знайти за обчисленням обраних для вимірювання показників, при

цьому не порівнюючи їх із відповідними показниками аналогічних засобів. Для цього було обрано низку параметрів: стійкість алгоритму, довжина секретного ключа, швидкість шифрування. Визначаємо величину параметрів якості в балах та встановлюємо граничні значення (гірші, кращі, середні). Дані для кожного параметра представлено у таблиці 5.3.

Таблиця 5.3 – Основні параметри методу блокового шифрування

Параметри	Абсолютне значення параметру			Коефіцієнт вагомості параметру
	Краще +5…+4	Середнє +3	Гірше +1…+2	
Стійкість алгоритму	4			0,5
Довжина секретного ключа		3		0,3
Швидкість шифрування		3		0,2

Із врахування коефіцієнтів вагомості відповідних параметрів можна визначити абсолютний рівень якості інноваційного рішення за формулою:

$$K_{я.а.} = \sum_{i=1}^{n} Рн_i * a_i, \text{ де} \quad (5.1)$$

$Рн_i$ – числове значення і-го параметра інноваційного рішення, n – кількість параметрів інноваційного рішення, що прийняті для оцінювання, a_i – коефіцієнт вагомості відповідного параметра (сума коефіцієнтів вагомості всіх параметрів повинна дорівнювати 1). Таким чином, абсолютний рівень якості методу блокового шифрування файлів складає 3.5 бали. Тепер потрібно визначити відносний рівень якості та порівняти показники з абсолютними показниками якості найліпших аналогів (табл. 5.4).

Таблиця 5.4 – Порівняння основних параметрів методу блокового шифрування

Параметри	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметру
	Базовий	Новий		
Стійкість алгоритму	0,4	0,8	2	0,6
Довжина секретного ключа	128Б	128Б	1	0,3
Швидкість шифрування	0.15	0.2	0.75	0,2

Відносний рівень якості мето методу блокового шифрування визначаємо за формулою 5.2.

$$K_{я.в.} = \sum_{i=1}^{n} q_i * a_i, \text{ де} \quad (5.2)$$

Відносний рівень для методу псевдонедетермінованого блокового шифрування файлів становить 1.65 – тобто розроблений метод якісніший на 65% за аналогічні.

Конкурентоспроможність визначається критеріями оцінювання споживачів конкретного ринку. Для розрахунку конкурентоспроможності необхідно сформувати таблицю 5.5, із відповідними показниками.

Таблиця 5.5 – Нормативні та економічні параметри методу псевдонедетермінованого блокового шифрування файлів.

Параметри	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметру
	Базовий	Новий		
Стійкість алгоритму	0,4	0,8	2	0,6
Довжина секретного ключа	128Б	128Б	1	0,3
Швидкість шифрування	0,15	0,2	0,75	0,2
Ціна за продукт	14000	10000	0,71	-

Загальний показник конкурентоспроможності (К) визначається з урахуванням вищевказаних параметрів за формулою 5.3.

$$K = \frac{I_{Т.П.}}{I_{Е.П.}} = \frac{1,65}{0,83} = 2, \quad (5.3)$$

де $I_{Т.П.}$ – індекс технічних параметрів (відносний рівень якості інноваційного рішення); $I_{Е.П.}$ – індекс економічних параметрів розрахований нижче за формулою 5.4:

$$I_{Е.П.} = \frac{РН_{ЕІ}}{РБ_{ЕІ}} = \frac{10000}{14000} = 0,71 \quad (5.4)$$

де $РН_{ЕІ}$, $РБ_{ЕІ}$ – економічні параметри (ціна придбання та споживання товару) відповідно нового та базового товарів.

Відповідно до розрахунків загальний показник конкурентоспроможності становить 2, що свідчить про більшу конкурентоспроможність аніж в порівнянні з аналогом.

5.2 Прогнозування витрат на виконання науково-дослідної та конструкторсько-технологічної роботи

5.2.1 Розрахунок витрат, що стосуються виконавців розробки методу та засобу автентифікації

Команда розробки методу та засобу псевдонедетермінованого блокового шифрування файлів складається з керівника, інженера-програміста та тестувальника.

Основна заробітна плата для розробників (дослідників) *Зо*, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_О = \frac{M}{T_p} t \qquad (5.5)$$

де M – місячний посадовий оклад розробника;

T_p – кількість робочих днів у місяці, $T_p = 22$ дні; t – число днів роботи.

Розрахунки заробітної плати для розробників наведені в таблиці 5.6

Таблиця 5.6 – Розрахунки основної заробітної плати розробників

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Керівник	25000	1136,36	7	7945,52
Розробник	20000	909,1	13	11818,3
Тестувальник	15000	681,82	3	2045,46
Всього:				21809,28

Основна заробітна плата робітників *Зр*, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$З_Р = \sum_{i=1}^{n} t_i \cdot C_i \,, \qquad (5.6)$$

де t_i – норма часу (трудомісткість) на виконання конкретної роботи, годин; n – число робіт по видах та розрядах; C_i – погодинна тарифна ставка робітника

відповідного розряду, який виконує дану роботу. C_i визначається за формулою:

$$C_i = \frac{M_M * K_i}{T_p * T_{ЗМ}}, \quad (5.7)$$

де M_M – розмір мінімальної заробітної плати за місяць, грн.; в 2021 році мінімальна заробітна плата становить – 6500 грн., K_i – тарифний коефіцієнт робітника відповідного розряду, T_p = 22 дні; $T_{зм}$ – тривалість зміни, $T_{зм}$ =8 годин.

Таблиця 5.7 – Заробітна плата робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Проектування	8	7	69,8	1,89	558,4
Розробка	9	5	56,87	1,54	511,83
Тестування	4	3	49,86	1,35	187,44
Всього					1257,67

Додаткова заробітна плата $З_Д$ всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_Д = 0.1 * (З_О + З_Р) = 0{,}1 * (21809{,}28 + 1257{,}67) = 2306{,}69 \text{ (грн.)} \quad (5.8)$$

Нарахування на заробітну плату $Н_{ЗП}$ розраховується як 22% від суми основної та додаткової заробітної плати:

$$Н_{ЗП} = (З_О + З_Р + З_Д) \times \frac{\beta}{100} = (21809{,}28 + 1257{,}67 + 2306{,}69) * 0{,}22 = 5582{,}2 \text{ (грн.)} \quad (5.9)$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_р$ – основна заробітна плата робітників, грн.;

$З_д$ – додаткова заробітна плата розробників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне страхування.

Розрахунок амортизаційних відрахувань виконується за такою формулою:

$$A = \frac{Ц}{t_B} \times \frac{T}{12} \quad (5.10)$$

де $Ц$ – балансова вартість обладнання, грн;

T – термін використання (T=22 дні= 0,73 місяців);

t_B – корисний час використання (t_B для комп'ютера становить 4 роки).

Під час виконання розробки використовувався ноутбук вартістю 40000 грн. Амортизаційні відрахування для персонального комп'ютера представлені у таблиці 5.8.

Таблиця 5.8 - Амортизаційні відрахування

Найменування	Ціна, грн.	Корисний час використання, роки	Термін використання, міс.	Сума амортизації, грн.
ПК	34000	4	0,73	517,08
Всього	517,08			

Витрати на силову електроенергію розраховуються за формулою:

$$B_E = B \times П \times Ф \times K_П \quad (5.11)$$

де B – вартість 1кВт-години електроенергії (В=4,62 грн/кВт);

$П$ – установлена потужність комп'ютеру ($П$=0,74 кВт);

$Ф$ – фактична кількість годин роботи комп'ютеру ($Ф$=22*8=176 год);

$K_П$ – коефіцієнт використання потужності ($K_П$ <1, $K_П$ = 0,8).

Відповідно до формули 5.11 витрати на силову електроенергію:

$$B_E = 4{,}62 \times 0{,}74 \times 176 \times 0{,}8 = 481{,}36 \text{ (грн.)}$$

Інші витрати $B_{ін}$ можна прийняти як (100-300)% від суми основної заробітної плати розробників, які виконували роботу, тобто:

$$B_{ін} = 1 \cdot (\ 21809{,}28 + 1257{,}67\ ) = 23066{,}95 \text{ (грн.)} \quad (5.11)$$

Сума усіх попередніх витрат дає загальні витрати на виконання роботи. Усі витрати складають:

$$B = 21809{,}28 + 1257{,}67 + 2306{,}69 + 5582{,}2 + 481{,}36 + 23066{,}95 + 517{,}08 = 55021{,}23$$

(грн.)

Розрахунок загальної вартості наукової розробки $В_{заг}$ за формулою:

$$B_{заг} = \frac{B}{\alpha}, \quad (5.12)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відносних одиницях.

$$B_{заг} = \frac{55021{,}23}{1} = 55021{,}23 \text{ (грн.)}$$

Прогнозування загальних витрат *ЗВ* на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{B_{заг}}{\beta} \quad (5.13)$$

Розрахунок прогнозованих загальних витрат:

$$ЗВ = \frac{55021{,}23}{0{,}7} = 78601{,}76 \text{ (грн.)}$$

5.2.2 Розрахунок собівартості розробки методу та засобу шифрування

Витрати на силову електроенергію розраховуються за формулою:

$$B_E = B \times П \times Ф \times K_П \quad (5.14)$$

де *В* – вартість 1кВт-години електроенергії (В=4,62 грн/кВт);

П – установлена потужність комп'ютеру (*П*=0,74 кВт);

Ф – фактична кількість годин роботи комп'ютеру (*Ф*=22*8=176 год);

$K_П$ – коефіцієнт використання потужності ($K_П$ <1, $K_П$ = 0,8).

Відповідно до формули 5.14 витрати на силову електроенергію:

$$B_E = 4{,}62 \times 0{,}74 \times 176 \times 0{,}8 = 481{,}36 \text{ (грн.)}$$

Основна заробітна плата робітників *Зр*, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$З_Р = \sum_{i=1}^{n} t_i \cdot C_i \, , \quad (5.15)$$

де t_i – норма часу (трудомісткість) на виконання конкретної роботи, годин; n – число робіт по видах та розрядах; C_i – погодинна тарифна ставка робітника відповідного розряду, який виконує дану роботу. C_i визначається за формулою:

$$C_i = \frac{M_M * K_i}{T_p * T_{ЗМ}}, \quad (5.16)$$

де M_M – розмір мінімальної заробітної плати за місяць, грн.; в 2021 році мінімальна заробітна плата становить – 6500 грн., K_i – тарифний коефіцієнт робітника відповідного розряду, T_p = 22 дні; $T_{зм}$ – тривалість зміни, $T_{зм}$ =8 годин.

Таблиця 5.9 – Заробітна плата робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Розробка	8	5	56,875	1,54	455
Тестування	8	4	53.55	1,45	428,4
Впровадження	2	2	40,25	1,09	80,51
Всього					963,91

Додаткова заробітна плата $З_Д$ всіх робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_Д = 0.1 * (З_Р) = 0{,}1 * (963{,}91) = 96{,}39 \ (\text{грн.}) \quad (5.17)$$

Нарахування на заробітну плату $Н_{ЗП}$ розраховується як 22% від суми основної та додаткової заробітної плати:

$$Н_{ЗП} = (З_Р + З_Д) \times \frac{\beta}{100} = (963{,}91 + 96{,}39) * 0{,}22 = 233{,}27 \ (\text{грн.}) \quad (5.18)$$

де $З_р$ – основна заробітна плата робітників, грн.;

$З_Д$ – додаткова заробітна плата робітників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне страхування.

Загальновиробничі витрати з рахунку на одиницю продукції можна розрахувати за нормативами відносно до основної заробітної плати основних робітників, які виготовляють продукцію :

$$ЗВВ = Н_В * З_О, \quad (5.19)$$

Норматив загальновиробничих витрат для програмних продуктів становить 230-270%.

$$ЗВВ = 2,7 * 963,91 = 2602,56 \text{ (грн.)}$$

Сума попередніх витрат утворює виробничу собівартість розробки.

$$S_В = 481,36 + 963,91 + 96,39 + 233,27 + 2602,56 = 4377,5 \text{ (грн.)} \quad (5.20)$$

5.3 Розрахунок мінімальної ціни та чистого прибутку від реалізації розробки методу та засобу псевдонедетермінованого блокового шифрування файлів

Ціна – це грошовий вираз вартості товару (продукції, послуги). Вона завжди коливається навколо ціни виробництва (перетвореної форми вартості одиниці товару, що дорівнює сумі витрат виробництва й середнього прибутку) та відображає рівень суспільне необхідних витрат праці.

Виходячи з того, що розробки, як правило, приймаються та впроваджуються за завданням замовника, або коли результатом розробки є продукція, що підлягає державному регулюванню, то нижню межу ціни реалізації розробки можна розрахувати за формулою 5.21:

$$Ц = S_В \cdot (1 + \frac{Р}{100}) \cdot (1 + \frac{\omega}{100}), \quad (5.21)$$

де $S_В$ – виробнича собівартість інноваційного рішення, грн.;

$Р$ – норматив рентабельності узгоджений із замовником або встановлений державою, (Р=30…60%);

ω – ставка податку на додану вартість, % (з осені 2021 року $\omega = 20\%$).

$$Ц = 4377,5 \cdot (1 + \frac{55}{100}) \cdot (1 + \frac{20}{100}) = 8142,15 \text{ (грн.)} \quad (5.21)$$

Із врахуванням коефіцієнта якості ціна розробки становить 28497,5 грн.

Чистий прибуток від реалізації розробки можна розрахувати за формулою:

$$П = (Ц - \frac{(Ц - МР)\cdot f}{100} - S_В - \frac{q\cdot S_В}{100})\cdot(1-\frac{h}{100})\cdot РП, \quad (5.22)$$

де *Ц* – ціна розробки, грн.; *МР* – вартість матеріальних та інших ресурсів, що були придбані виробником для виготовлення розробки (*МР*=(0,1…0,2) *Ц*), грн.;

f – зустрічна ставка податку на додану вартість, %; $S_В$ – виробнича собівартість розробки, грн.; *q* – норматив, який визначає величину адміністративних витрат, витрат на збут та інші операційні витрати, % (рекомендовано *q*=5…10%); *h* – ставка податку на прибуток, %, *РП* – прогнозований попит продажів.

$$П = (28497{,}5 - \frac{(28497{,}5 - 28497{,}5\cdot 0{,}2)\cdot 14}{100} - 4377{,}5 - \frac{6\cdot 4377{,}5}{100})\cdot(1-\frac{18}{100})\cdot 2 = 35853{,}7 \text{ (грн.)} \quad (5.23)$$

Прогнозований чистий прибуток від реалізації розробки складає 35853 грн.

5.4 Розрахунок терміну окупності коштів вкладених у наукову розробку методу та засобу шифрування файлів

Термін окупності вкладених у реалізацію наукового проекту інвестицій розраховано за формулою 5.24:

$$T_{ОК} = \frac{ЗВ}{П} = \frac{78601{,}76}{35853{,}7} = 2{,}19 \text{ (роки)} \quad (5.24)$$

Оскільки $T_{ОК} < 3$ років, то фінансування наукової розробки методу та засобу псевдонедетермінованого блокового шифрування файлів є доцільним.

5.5 Висновки до розділу

Отже, у цьому розділі виконано обґрунтування економічної доцільності проведення наукового дослідження та розробки методу, та засобу псевдонедетермінованого блокового шифрування файлів.

Рівень комерційного потенціалу розробки методу та засобу псевдонедетермінованого блокового шифрування файлів є вище середнього показника.

На основі параметрів засобу шифрування визначено абсолютний рівень якості методу та засобу блокового шифрування файлів, який складає 3,5 бали.

Відносний рівень якості розробки складає 1,65. Це означає, що нова розробка якісніша на 65% відносно товарів-аналогів. Загальний показник конкурентоспроможності становить 2, що свідчить про більшу конкурентну спроможність засобу псевдонедетермінованого блокового шифрування файлів у порівнянні з товаром-аналогом на 100%.

Загальні витрати, що стосуються виконавців розробки склали 78601,76 грн, а собівартість розробки – 4377,5 грн.

Розраховано мінімальну ціну та прогнозований чистий річний прибуток від реалізації розробки, які склали 28497,5 грн. та 35853,7 грн. відповідно. Термін окупності продукції вкладених інвестицій складає 2,19 роки, що свідчить про доцільність фінансування розробки.

ВИСНОВКИ

Було проаналізовано, що в сучасному світі більшість інформації обробляється, передається та зберігається у вигляді файлів. Переважна частина сучасних програмних засобів для роботи з файлами не передбачають процесу захисту даних. Отже, є необхідність використання сторонніх засобів для забезпечення необхідного рівня захисту інформації. Для захисту файлів зазвичай використовують різні алгоритми блокового шифрування, оскільки даний тип криптографічних перетворень показує найбільшу ефективність при роботі з даними довільного обсягу.

Під час виконання роботи було проведено дослідження щодо сучасних підходів блокового шифрування, режимів шифрування, основних компонент, які використовуються для шифрування. Проаналізувавши різні реалізації алгоритмів блокового шифрування бачимо, що для деяких уже знайдено вразливості, інші мають теоретично доведену можливість реалізації вразливості, також існує можливість задіяти недоліки алгоритмів при використанні спрощених або часткових реалізацій окремих криптографічних перетворень. Виходячи з цього було запропоновано розробити метод, а згодом його реалізацію в вигляді програмного засобу, який збільшить стійкість шифрування файлів для проведення лінійного, статистичного та диференційного криптоаналізу.

Отже, було запропоновано застосувати псевдонедетермінований підхід, для визначення алгоритму шифрування на кожному з раундів. Для цього виконано математичний опис підхожу блокового шифрування, в якому визначено його відмінності в порівнянні з аналогами. За основу було обрано дві структури блокового шифрування – SPN мережа та схема Lai-Massey, які відповідають сучасним стандартам шифрування.

Для забезпечення додаткової стійкості алгоритму шифрування також було розроблено процедуру розгортання раундових ключів на основі секретного ключа та значень незвідних поліномів.

Перед проведенням розробки засобу псевдонедетермінованого блокового шифрування файлів було описано його архітектурні компоненти та алгоритми процедур, які реалізують запропонований метод.

Виконано розробку засобу для реалізації алгоритму шифрування написаного на мові програмування Java. Дана мова була обрана оскільки має ряд переваг та є найбільш зручна для написання програм, які повинні працювати на серверному рівні. Також при проектуванні засобу було вирішено розробити алгоритм для спрощення взаємодії з користувачем. Для розв’язання основних поставлених задач при розробці засобу шифрування достатньо базових можливостей мови програмування, але для написання інтерфейсу та іншого додаткового функціоналу було використано популярні сучасні фреймворки, такі як Spring та Vaadin.

Для розробленого засобу шифрування було проведено тестування на відповідність до низки функціональних параметрів, які визначається для криптографічних перетворень типу генераторів псевдовипадкових значень, що є доцільним у використанні до алгоритму шифрування. Результати тестування показали, що отриманий засіб задовольняє поставлені вимоги та може бути використаний для стійкого шифрування файлів.

Також було проведено розрахунки економічної доцільності реалізації програмного засобу для псевдонедетермінованого блокового шифрування файлів. При порівнянні з аналогами засобів для шифрування файлів, визначено, що запропонований метод має показник якості, який перевищує конкурентів. На основі проведених розрахунків витрати для дослідження та розробки нового методу шифрування, визначено доцільність її проведення та термін окупності, який складає 2 роки та 2 місяці.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Why Information Security is Important?:.URL:https://www.infosecacademy.io/blog/importance-of-information-security/ (дата звертання: 15.09.2021).
2. Василенко В. С. , Матов О. Я. Теорія інформації та кодування : навч. посіб. Київ: НАН України, 2014. 439 с.
3. Корченко О.Г. Системи захисту інформації : монографія. Київ: НАУ, 2004. 264с.
4. Symmetric Cryptography: URL:https://www.sciencedirect.com/topics/computer-science/symmetric-cryptography (дата звернення: 20.09.2021).
5. Bellare M., Rogaway P. Introduction to Modern Cryptography : посібник. Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093. USA. 2005. 283 с.
6. NIST Special Publication 800-57 Part 1 Revision 3. Recommendation for Key Management – Part 1: General (Revised) [Чинний від 15-07-2007]. Вид офіц. США: Computer Security Division (Information Technology Lab), 2007. 143с. (Інформація та документація).
7. Shannon C. E. Communication Theory of Secrecy Systems: монографія. Bell System Technical Journal. 2012. 28с.
8. Advanced Encryption Standard (AES): URL:https://www.techtarget.com/searchsecurity/definition/Advanced-Encryption-Standard (дата звернення: 27.09.2021).
9. Single Key Recovery Attacks on 9-round Kalyna-128/256 and Kalyna-256/512: / Donghoon C. та ін. ; Indraprastha Institute of Information Technology, Delhi. 21с.
10. Specification of Camellia – a 128-bit Block Cipher: монографія: / Aoki K. та ін., видання 2-ге; Mitsubishi Electric Corporation. 2001. 35с.
11. Courtois N. Security Evaluation of GOST 28147-89 In View Of International Standardisation: монографія: University College London, Gower Street, London, UK. 2011. 15с.

12. Modugu R., Yong-Bin K., Minsu C. Design and performance measurement of efficient IDEA: монографія: Austin, TX, USA. 2010.
13. Improved Cryptanalysis of the Block Cipher KASUMI: монографія / Keting J. та ін.; Institute for Advanced Study, Tsinghua University, China. 2013. 12с.
14. Ashly J., Panchami V. An Analysis on Low Cost and Performance of Hardware and Software Oriented Lightweight Block Ciphers for IoT Applications: монографія: Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021. 2021. 10с.
15. Changyong P., Chuangying Z., Yuefei Z., Fei K. Improved side channel attack on the block cipher NOEKEON: монографія: School of Computer and Control, Guillin University of Electronic Technology, Guilin, China. 2012. 9с.
16. Biham E., Dunkelman O., Keller N. Differential-Linear Cryptanalysis of Serpent: монографія: IST Programme under Contract IST-1999-12324. 2021.
17. Jiyan Z., Ting C., Chenhui J. Structural Attack on Reduced-Round Skipjack: монографія: Information Science and Technology Institute, Zhengzhou 450000, China. 2018. 8с.
18. Chujiao M., Chandy J., Zhijie S. Algebraic Side-Channel Attack on Twofish: монографія: University of Connecticut, Storrs, Connecticut 06269, USA. 2017. 12с.
19. Sekar G., Mouha N., Velichkov V., Preneel B. Meet-in-the-Middle Attacks on Reduced-Round XTEA: монографія: Department of Electrical Engineering ESAT/SCD-COSIC, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001 Heverlee, Belgium. 18с.
20. Biryukov A. Product Cipher, Superencryption: монографія: FDEF, Campus LimpertsbergUniversity of LuxembourgLuxembourgLuxembourg. 2011.
21. Shannon Ciphers and Perfect Security: URL:https://www.cantorsparadise.com/shannon-ciphers-and-perfect-security-d70c10379ac2 (дата звернення: 13.10.2021).
22. Feistel H. Cryptography and Computer Privacy: посібник: Scientific American. 1973. 228.

23. Dawson E., Gustafson H., Pettitt A. Strict Key Avalanche Criterion: монографія: formation Security Research Centre Queensland University of Technology. 1992. 8с.
24. Feistel cipher: URL: https://cryptography.fandom.com/wiki/Feistel_cipher (дата звернення: 17.10.2021).
25. Preneel B., Rijmen V., Bosselaers A. Principles and Performance of Cryptographic Algorithms // Dr. Dobb's Journal, Vol. 23, No. 12, 1998. С. 126-131.
26. An overview on cryptanalysis of ARX ciphers: URL: https://crypto.polito.it/content/download/480/2850/file/document.pdf (дата звернення: 23.10.2021).
27. Lai X., Massey J. A Proposal for a New Block Encryption Standard: монографія: Institute for Signal and Information Processing Swiss Federal Institute of Technology CH, Zurich, Switzerland. 1991. 16с.
28. Wesley W., Weldon E. Error-correcting codes: монографія: The mit press Cambridge, Massachusetts, and London, England. 1972. 595с.
29. Comparing Python to Other Languages: URL: https://www.python.org/doc/essays/comparisons/ (дата звернення: 28.10.2021).
30. Overview of C#: URL: https://www.wideskills.com/csharp/overview-csharp (дата звернення: 28.10.2021).
31. How Is Java Different From Other Languages? : URL: https://compscicentral.com/how-is-java-different-from-other-languages/ (дата звернення: 29.10.2021).
32. Introduction to Spring Framework: URL: https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html (дата звернення: 02.11.2021).
33. NIST Special Publication 800-22 Revision 1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Application [Чинний від 15-04-2010]. Вид офіц. США: Computer Security Division (Information Technology Lab), 2010. 131с. (Інформація та документація).

34. What is Software Testing? Definition, Basics & Types in Software Engineering: URL: https://www.guru99.com/software-testing-introduction-importance.html (дата звернення: 12.11.2021).
35. International Standard for Systems and Software Engineering. 1059-1993 - IEEE Guide for Software Verification and Validation Plans. [Чинний від 28-04-1994]. ISO/IEC/IEEE 24748-5:2017(E), 1994.
36. Юнит-тестирование: URL: https://habr.com/ru/post/169381/ (дата звернення: 16.11.2021).
37. Unit Tests, How to Write Testable Code and Why it Matters: URL: https://www.toptal.com/qa/how-to-write-testable-code-and-why-it-matters (дата звернення: 18.11.2021).
38. JUnit 5 User Guide: URL: https://junit.org/junit5/docs/current/user-guide/ (дата звернення: 22.11.2021).

ДОДАТКИ

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ д. т. н.,проф.
_____ В. А. Лужецький
«____» ____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання магістерської кваліфікаційної роботи

на тему: «МЕТОД ТА ЗАСІБ ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ»
08-20.МКР.012.00.000 ТЗ

Керівник магістерської кваліфікаційної роботи
к.т.н., доц. каф. ЗІ. Ю. В. Баришев

Вінниця 2021

1 Підстави для проведення робіт

Робота проводиться на підставі наказу ректора ВНТУ від 1 вересня 2021 року № 207.

Дата початку роботи 09.09.21 р.

Дата закінчення роботи 20.12.21 р.

2 Мета та призначення МКР

Мета – покращення існуючих підходів до шифрування файлів.

Об'єктом дослідження є процес забезпечення стійкого шифрування файлів.

Предметом є методи блокового шифрування, а також процес псевдонедетермінованості, як підхід до шифрування.

Актуальність теми – розробка більш стійких методів блокового шифрування є затребуваною виходячи з аналізу сучасних стандартів блокового шифрування, які в більшості випадків є морально застарілими та мають певні вразливості до різних методів криптоаналізу.

Використання двох структур для виконання процесу шифрування додає елемент псевдонедетермінованості, натомість швидкість шифрування не змінюється. Такий підхід дозволяє підвищити стійкість до лінійного та диференційного криптоаналізу. Відповідно для успішного криптоаналізу необхідно буде витрати в два рази більше ресурсів та часу, оскільки використовується два різних підходи одночасно.

3 Вихідні дані для проведення МКР

МКР проводиться вперше і вихідними даними для проведення МКР є:

3.1 Василенко В. С., Матов О. Я. Теорія інформації та кодування : навч. посіб. Київ: НАН України, 2014. 439 с.

3.2 Корченко О.Г. Системи захисту інформації: монографія. Київ: НАУ, 2004. – 264с.

3.3 Symmetric Cryptography.: URL:https://www.sciencedirect.com/topics/computer-science/symmetriccryptography (дата звернення: 15.09.2021).

3.4 Bellare M., Rogaway P. Introduction to Modern Cryptography : посібник. Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093, USA. 2005. 283с.

3.5 Shannon C. E. Communication Theory of Secrecy Systems: // Bell System Technical Journal. 2012. 28с.

4 Виконавці МКР

Студент групи 1 БС-20 м Ціхоцький Микита Сергійович

6 Вимоги до виконання МКР

Для покращення стійкості процесу шифрування файлів за рахунок введення процесу псевдонедетермінованості необхідно розв'язати такі задачі:

- проаналізувати відомі методи блокового шифрування;
- проаналізувати режими роботи блокових шифрів;
- виконати узагальнений математичний опис процесу шифрування;
- розробити метод псевдонедетермінованого блокового шифрування;
- розробити алгоритм роботи програмного засобу;
- реалізувати програмний засіб для шифрування файлів на основі розробленого методу;
- провести тестування методу шифрування;
- виконати розрахунки економічної доцільності розробки.

6 Вимоги до супровідної документації

Графічна і текстова документація повинна відповідати діючим стандартам України – ДСТУ 3008:2015.

7 Етапи МКР

Робота з теми виконується у 8 етапів.

Зміст етапу	Початок – закінчення	Очікувані результати	Звітна документація
Аналіз завдання. Вступ	09.09.2021 – 14.09.2021	Вступ	Чернетка вступу
Розробка технічного завдання	15.09.2021 –16.09.2021	Технічне завдання	Проект технічного завдання
Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	17.09.2021 – 05.10.2021	Аналіз існуючих аналогів. Вибір напрямку дослідження Аналіз відомих методів. Постановка завдання	Чернетка першого розділу
Удосконалення методу шифрування файлів	06.10.2021 – 19.11.2021	Математичний опис методу шифрування. Запропонований метод шифрування.	Чернетка другого розділу
Експериментальні дослідження	20.11.2021 – 28.11.2021	Програмний засіб, який реалізує розроблювані методи	Чернетка третього розділу
Розробка економічного розділу	29.11.2021 – 10.12.2021	Економічні показники дослідження	Чернетка з економічного розділу
Оформлення пояснювальної записки	11.12.2021 – 17.12.2021	Пояснювальна записка	Пояснювальна записка

8 Очікувані результати та порядок реалізації МКР

Передбачається розробка нового методу шифрування, який спрямований на покращення стійкості шифрування файлів. Заплановане створення програмного засобу, який може бути використаний у забезпечення процесу шифрування файлів.

9 Матеріали які подаються після закінчення МКР

По завершенню роботи подається пояснювальна записка та ілюстративна частина.

10 Порядок приймання МКР та її етапів

Апробація на науково-технічних конференціях та семінарах. Результати роботи будуть розглядатися на засіданні ДЕК із захисту магістерських кваліфікаційних робіт.
Попередній захист та доопрацювання МКР – 3-4 грудня 2021 р.
Представлення МКР до захисту – 20 грудня 2021 р.
Захист МКР – 23 грудня 2021 р.

11 Вимоги до розроблення документації

Документація буде виконуватись за допомогою комп’ютерного набору у відповідності вимог ДСТУ 3008:2015 «Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання».

12 Вимоги щодо технічного захисту інформації з обмеженим доступом

У зв’язку з тим, що дана робота не містить інформації, що потребує захисту у відповідності до законів України, заходи з її технічного захисту не передбачаються.

Додаток Б. Код засобу

```
@Route("")
public class MainView extends VerticalLayout {

   @Getter
   byte[] fileBytes;
   Cipher  cipher;
   File file;

   public MainView() {
      KeyTransformations keyTransformations = new KeyTransformations();
      try {
        cipher = Cipher.getInstance("AES");
      } catch (NoSuchAlgorithmException | NoSuchPaddingException e) {
         e.printStackTrace();
      }


      Button encrypt = new Button("Encrypt");
      encrypt.setMinWidth(250, Unit.PIXELS);
      encrypt.setMinHeight(100, Unit.PIXELS);
      encrypt.addClickListener(click -> {
         final Key key = new SecretKeySpec(keyTransformations.getSecretKey().getBytes(),"AES");
         try {
            cipher.init(Cipher.ENCRYPT_MODE, key);
         } catch (InvalidKeyException e) {
            e.printStackTrace();
         }
         byte[] encrypted = new byte[0];
         try {
            encrypted = cipher.doFinal(fileBytes);
         } catch (IllegalBlockSizeException | BadPaddingException e) {
            e.printStackTrace();
         }
         System.out.println("Encrypted: " + (Base64.getEncoder().encodeToString(encrypted)));
         file = new File("outputFile.txt");
         try (FileOutputStream outputStream = new FileOutputStream(file)) {
            outputStream.write(encrypted);
         } catch (IOException e) {
            e.printStackTrace();
         }
      });

      Button decrypt = new Button("Decrypt");
      decrypt.setMinWidth(250, Unit.PIXELS);
      decrypt.setMinHeight(100, Unit.PIXELS);

      MemoryBuffer memoryBuffer = new MemoryBuffer();

      Upload keyFileUpload = new Upload(memoryBuffer);
      keyFileUpload.addSucceededListener(event -> {
```

```
        InputStream fileData = memoryBuffer.getInputStream();
        keyTransformations.setSecretKey( new BufferedReader(new InputStreamReader(fileData))
                .lines().collect(Collectors.joining("\n")));
    });

    Upload fileUpload = new Upload(memoryBuffer);
    fileUpload.addSucceededListener(event -> {
        InputStream fileData = memoryBuffer.getInputStream();
        try {
            fileBytes = addition(IOUtils.toByteArray(fileData));
        } catch (IOException e) {
            e.printStackTrace();
        } });

    Button download = new Button("Download");
    download.setMinWidth(250, Unit.PIXELS);
    download.setMinHeight(100, Unit.PIXELS);

    add(
            new H2("Welcome to the new way in block cipher cryptography."),
            new H2("Here you can encrypt and decrypt your important files and only users which may access
to the private key could get real information."),
            new HorizontalLayout(
                    new VerticalLayout(
                            new H4("Secret key file:"),
                            keyFileUpload
                    ),
                    new VerticalLayout(
                            new H4("File for transformations:"),
                            fileUpload
                    ) ),
            new HorizontalLayout(
                    encrypt,
                    decrypt,
                    download
            ) );    }

  private byte[] addition(byte[] plainText) {
    int lengthDifference = plainText.length % 4;
    if (lengthDifference != 0) {
        List<Byte> data = new ArrayList<>();
        for (byte b : plainText) {
            data.add(b);
        }
        for (int i = 0; i < lengthDifference; i++) {
            data.add((byte) 0);
        }
        byte[] converted = new byte[data.size()];
        for (int i = 0; i < converted.length; i++) {
            converted[i] = data.get(i);
        }
        plainText = converted.clone();
    }
```

```
    return plainText;
  } }
```

Додаток В. Код тестування

```
@SpringBootTest
class BlockCipherApplicationTests {
        @Test
        void keyLengthTest() {
                KeyTransformations key = new KeyTransformations();
                String keyValue = key.getSecretKey();
                StringBuilder result = new StringBuilder();
                char[] chars = keyValue.toCharArray();
                for (char aChar : chars) {
                        result.append(
                        String.format("%8s", Integer.toBinaryString(aChar)).replaceAll(" ", "0");
        }
                assertEquals(128,result.length());
        }
        @Test
        void monobitFrequencyTest(){
                String round = Tests.convertStringToBinary(Tests.test);
                System.out.println(round);
                int ctr = Tests.onesCounter(round);
                System.out.println(ctr);
                double resultValue = Tests.pValue(ctr, round.length());
                System.out.println(resultValue);
                if (resultValue > 0.01){
                        assertTrue(true);
                } else fail();
        }
        @Test
        void monobitBlockFrequencyTest(){
                int ctr = Tests.onesCounter(Tests.binaryString);
                System.out.println(ctr);
```

```
                double resultValue = Tests.pbValue(ctr, 100);
                System.out.println(resultValue);
                if (resultValue > 0.01){
                        assertTrue(true);
                } else fail();
        }
        @Test
        void runsTest(){
                int ctr = Tests.onesCounter(Tests.binaryString);
                System.out.println(ctr);
                double resultValue = Tests.prValue(ctr, 100);
                System.out.println(resultValue);
                if (resultValue > 0.01){
                        assertTrue(true);
                } else fail();
        }

public class Tests {
   public static String convertStringToBinary(String input) {
      StringBuilder result = new StringBuilder();
      char[] chars = input.toCharArray();
      for (char aChar : chars) {
         result.append(
               String.format("%8s", Integer.toBinaryString(aChar)).replaceAll(" ", "0");
      }
      return result.toString();
   }
   public static int onesCounter(String input){
      int counter = 0;
      for (int i = 0; i < input.length(); i++){
         if(input.charAt(i) == '1'){
            counter+=1;
         }}
```

```
    return counter;
}
public static  double pValue(final int ones, final int n) {
    final int sum = (ones << 1) - n;
    final double pValue = erfc(Math.abs(sum)
            / (Math.sqrt(2) * Math.sqrt(n)));
    return pValue;
}
public static double pbValue(final int ones, final int n) {
    final int blockSize = (int) Math.sqrt(n);
    System.out.println("blockSize = " + blockSize);
    final int blockCount = n / blockSize;
    System.out.println("blockCount = " + blockCount);
    Fraction sum = Fraction.ZERO;
    for (int i = 0; i < blockCount; i++) {
        final Fraction p = new Fraction(ones, blockSize);
        final Fraction f = p.subtract(Fraction.ONE_HALF);
        sum = f.multiply(f).add(sum);
    }
    System.out.println("sum = " +sum);
    final double xSquared = new Fraction(4).multiply(
            new Fraction(blockSize)).multiply(sum).doubleValue();
    System.out.println("xSquared = " + xSquared);
    igamc(blockCount / 2.0, xSquared / 2.0);
    return 0.706438;
}
public static double prValue(final int e, final int n) {
    final int[] onesAndVn = onesAndVn(e, n);
    final int ones = onesAndVn[0];
    if (pValue(ones, n) < 0.01) {
        return 0.0;
    }
    final int doubleOnesAndZeroes = 2 * ones * (n - ones);
```

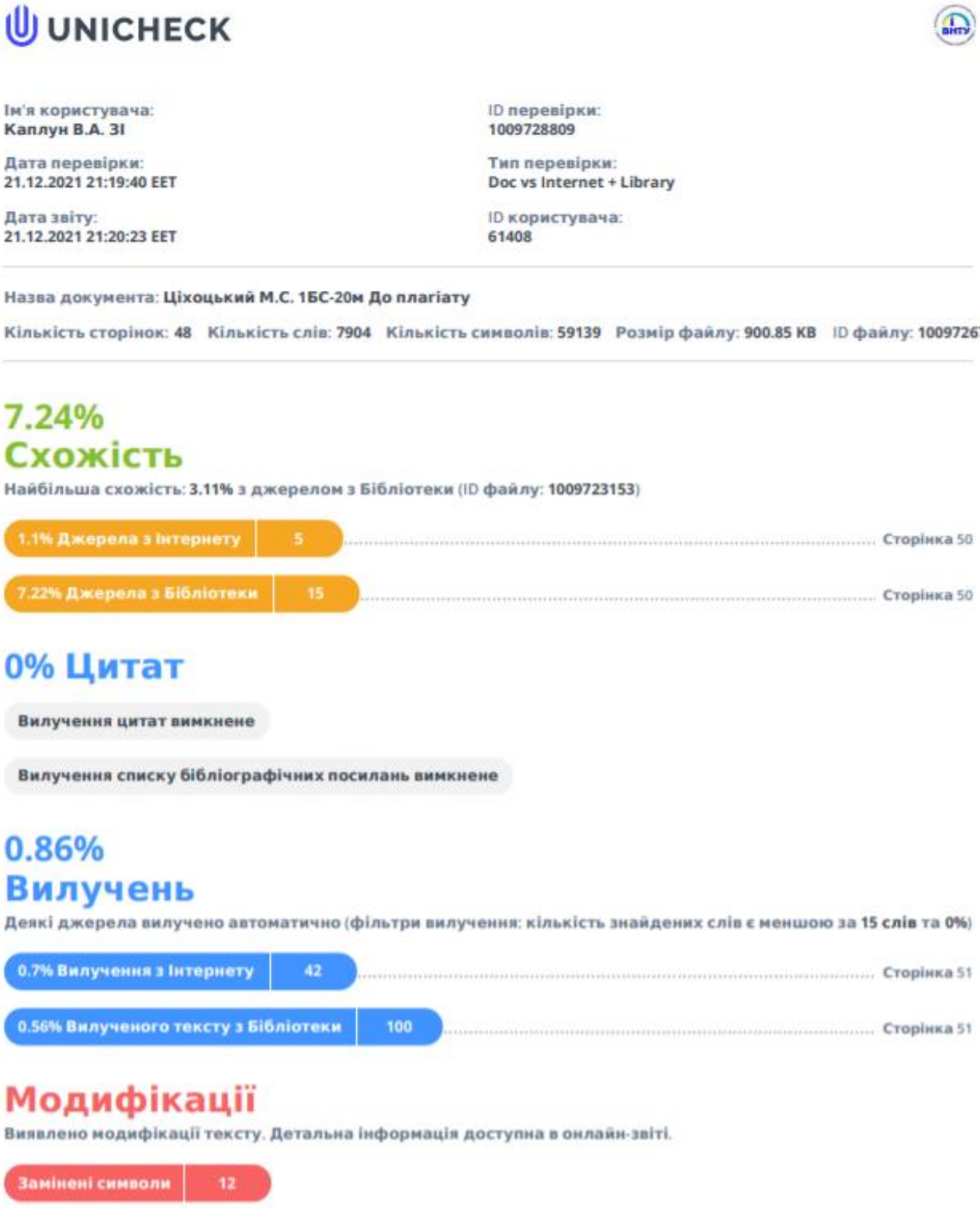
```
    final double pValue = erfc(new Fraction(onesAndVn[1] * n
            - doubleOnesAndZeroes, 2 * doubleOnesAndZeroes).abs()
            .doubleValue()
            * Math.sqrt(2 * n));
    return pValue;
}
private static int[] onesAndVn(final int e, final int n) {
    Random random = new Random();
    int ones = 0;
    int vN = 1;
    if (n > 0) {
        boolean last = random.nextBoolean();
        if (last) {
            ones++; }
        for (int i = 1; i < n; i++) {
            final boolean next = random.nextBoolean();
            if (next) {
                ones++;  }
            if (last != next) {
                vN++; }
            last = next  }   }
    return new int[] { ones, vN }   }
public static double igamc(final double a, final double x) {
        return Gamma.regularizedGammaQ(a, x);
}
public static  double erfc(final double x) {
        return 1 - Erf.erf(x);
}}
```

Додаток Г. Критерії оцінювання потенціалу розробки

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Критерій	1	2	3	4	5
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Критерій	1	2	3	4	5
		наявних фахівців			
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Додаток Д. Результати перевірки роботи на плагіат



UNICHECK

Ім'я користувача:
Каплун В.А. ЗІ

ID перевірки:
1009728809

Дата перевірки:
21.12.2021 21:19:40 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
21.12.2021 21:20:23 EET

ID користувача:
61408

Назва документа: Ціхоцький М.С. 1БС-20м До плагіату

Кількість сторінок: 48 Кількість слів: 7904 Кількість символів: 59139 Розмір файлу: 900.85 KB ID файлу: 1009726707

7.24%
Схожість

Найбільша схожість: 3.11% з джерелом з Бібліотеки (ID файлу: 1009723153)

1.1% Джерела з Інтернету 5 Сторінка 50

7.22% Джерела з Бібліотеки 15 Сторінка 50

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0.86%
Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 15 слів та 0%)

0.7% Вилучення з Інтернету 42 Сторінка 51

0.56% Вилученого тексту з Бібліотеки 100 Сторінка 51

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 12

Назва документа: **Ціхоцький М.С. 1БС-20м До плагіату**

ID файлу: **1009726707**

Схожість

Джерела з Інтернету 5

4	http://nozdr.ru/data/media/biblio/kolxoz/M/MV/MVsa/Crassidis%20J.,%20Junkins%20J.%20Optimal%20estimation%20of%20d...	0.46%
6	http://dspace.wunu.edu.ua/bitstream/316497/39773/1/%D0%9D%D0%B0%D0%B2%D1%87%D0%B0%D0%BB%D1%8C%D0%B...	0.33%
9	https://dspace.lboro.ac.uk/2134/9133	0.3%
11	http://altoma.cz/cz/stahnout-dokument/62/19/445__460-ridici_jednotka_.pdf	0.27%
15	http://www.calp.org.ar/info/revistas/Especiales/DISCAPACIDAD.pdf	0.24%

Джерела з Бібліотеки 15

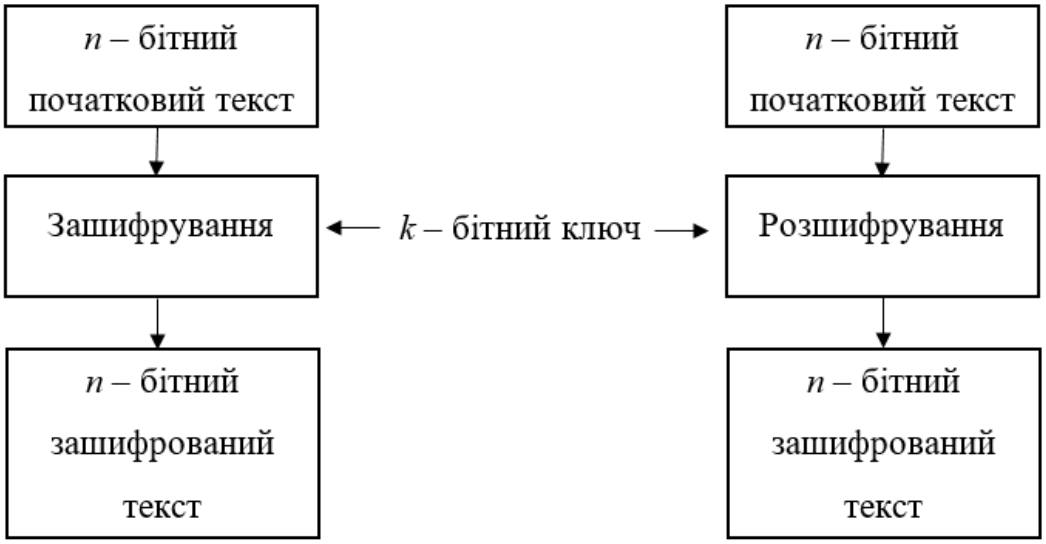
1	**Stashko_MKR 2012**	ID файлу: **1009723153**	Навчальний заклад: **Vinnytsia National Technical University**	3.11%
2	**ДП_2017_Мазур**	ID файлу: **2011424**	Навчальний заклад: **Vinnytsia National Technical University**	1.77%
3	**Ціхоцький БДР (до плагіату)**	ID файлу: **1004127281**	Навчальний заклад: **Vinnytsia National Technic...** 2 Джерела	1.3%
5	**1akit17b_rudych_plagiat**	ID файлу: **1008279043**	Навчальний заклад: **Vinnytsia National Technical University**	0.34%
7	**БДР_21_Говорун_на_плагіат**	ID файлу: **1008342321**	Навчальний заклад: **Vinnytsia National Technical University**	0.32%
8	**151_БевзІС_2018**	ID файлу: **3727535**	Навчальний заклад: **Vinnytsia National Technical University**	0.3%
10	**Розділи 6 і далі посібника по ФА**	ID файлу: **1001047028**	Навчальний заклад: **Vinnytsia National Technical Un...**	0.28%
12	**БДР_21_Казміревський**	ID файлу: **1008349366**	Навчальний заклад: **Vinnytsia National Technical University**	0.24%
13	**МКР_Мельничук_2018**	ID файлу: **5982256**	Навчальний заклад: **Vinnytsia National Technical University**	0.24%
14	**Лиськов**	ID файлу: **1009703289**	Навчальний заклад: **Vinnytsia National Technical University**	0.24%
16	**122БДР-БуксирЯО2021**	ID файлу: **1008312111**	Навчальний заклад: **Vinnytsia National Technical University**	0.23%
17	**БДР 21 Стойко М**	ID файлу: **1008393497**	Навчальний заклад: **Vinnytsia National Technical University**	0.22%
18	**141_ІльніцькийА.О._2021**	ID файлу: **1005686557**	Навчальний заклад: **Vinnytsia National Technical University**	0.22%

ІЛЮСТРАТИВНА ЧАСТИНА

МЕТОД ТА ЗАСІБ ПСЕВДОНЕДЕТЕРМІНОВАНОГО БЛОКОВОГО ШИФРУВАННЯ ФАЙЛІВ

(Назва магістерської кваліфікаційної роботи)

Схема шифрування та розшифрування за допомогою блокового шифру



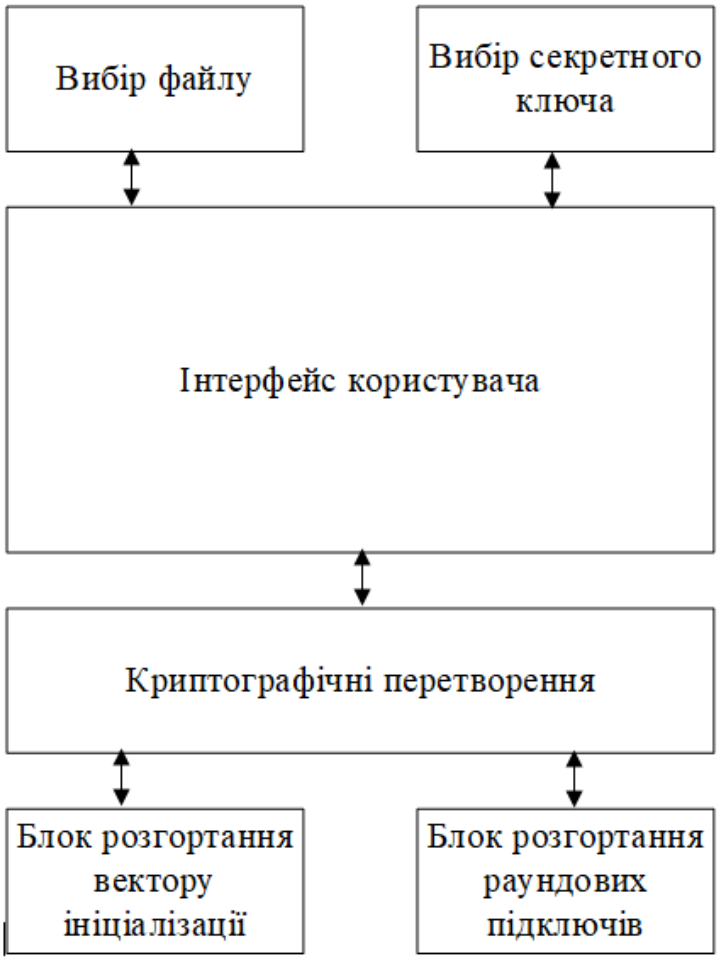
					08-20.МКР.012.00.000 І41			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Схема шифрування та розшифрування за допомогою блокового шифру	Літ.	Арк.	Аркушів
Перевір.		Баришев Ю.В.					1	1
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м		
Н. Контр.		Баришев Ю.В.						
Затверд.		Лужецький В.А.						

Порівняльна характеристика відомих алгоритмів блокового шифрування

Алгоритм шифрування	Розмір блоку [біт]	Ключ [біт]	Імплементація	Захищеність
AES	128	128, 192, 256	SW, HW	Стійкий
Kalyna	128	128, 192, 256	SW, HW	Стійкий
Camellia	128	128, 192, 256	SW, HW	Стійкий
GOST	64	256	HW	Зламаний
IDEA	64	128	SW	Відносно стійкий
KASUMI	64	128	SW, HW	Зламаний
KATAN	32, 48, 64	128	HW	Відносно стійкий
Noekeon	128	128, 256	SW, HW	Відносно стійкий
Serpent	128	128, 192, 256	SW, HW	Стійкий
Skipjack	64	80	SW, HW	Зламаний
Twofish	128	128, 192, 256	SW, HW	Стійкий
XTEA	64	128	SW, HW	Стійкий

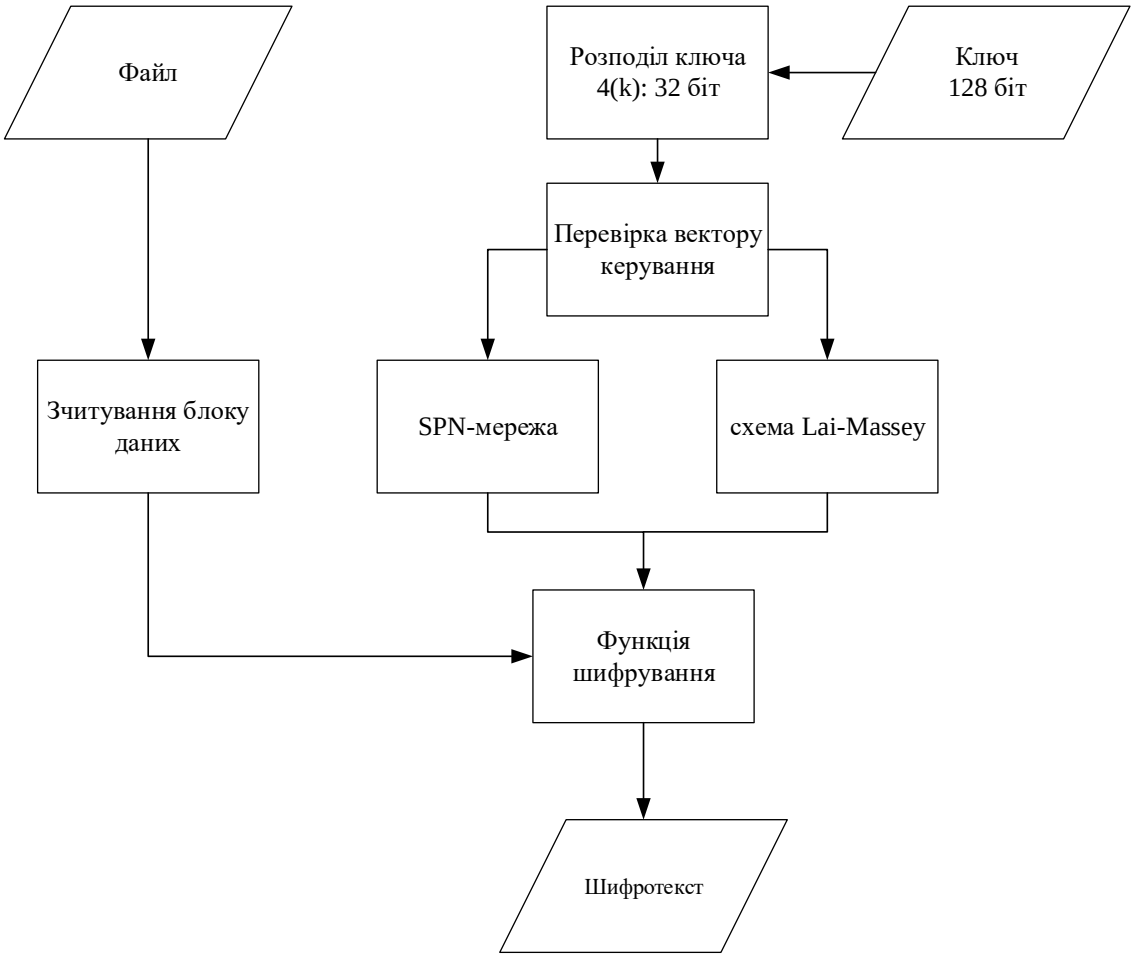
					08-20.МКР.012.00.000 І42		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Порівняльна характеристика відомих алгоритмів блокового шифрування	Літ.	Арк. 1 / Аркушів 1
Перевір.		Баришев Ю.В.					
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м	
Н. Контр.		Баришев Ю.В.					
Затверд.		Лужецький В.А.					

Узагальнена архітектура засобу блокового шифрування



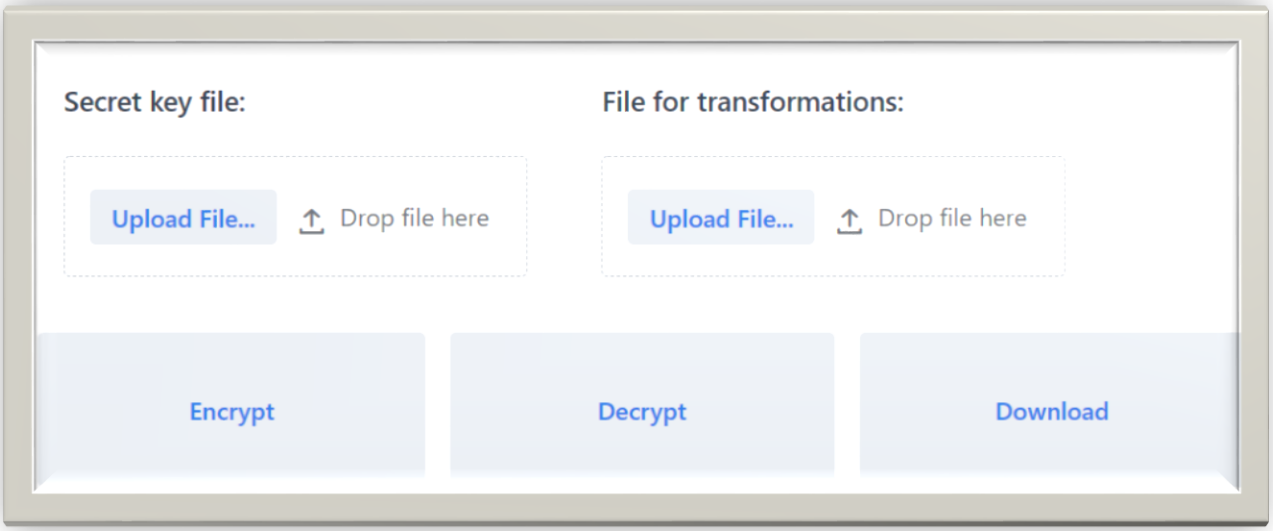
					08-20.МКР.012.00.000 ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Узагальнена архітектура засобу блокового шифрування	Літ.	Арк.	Аркушів
Перевір.		Баришев Ю.В.					1	1
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м		
Н. Контр.		Баришев Ю.В.						
Затверд.		Лужецький В.А.						

Структурна схема псевдонедетермінованого блокового шифрування файлів



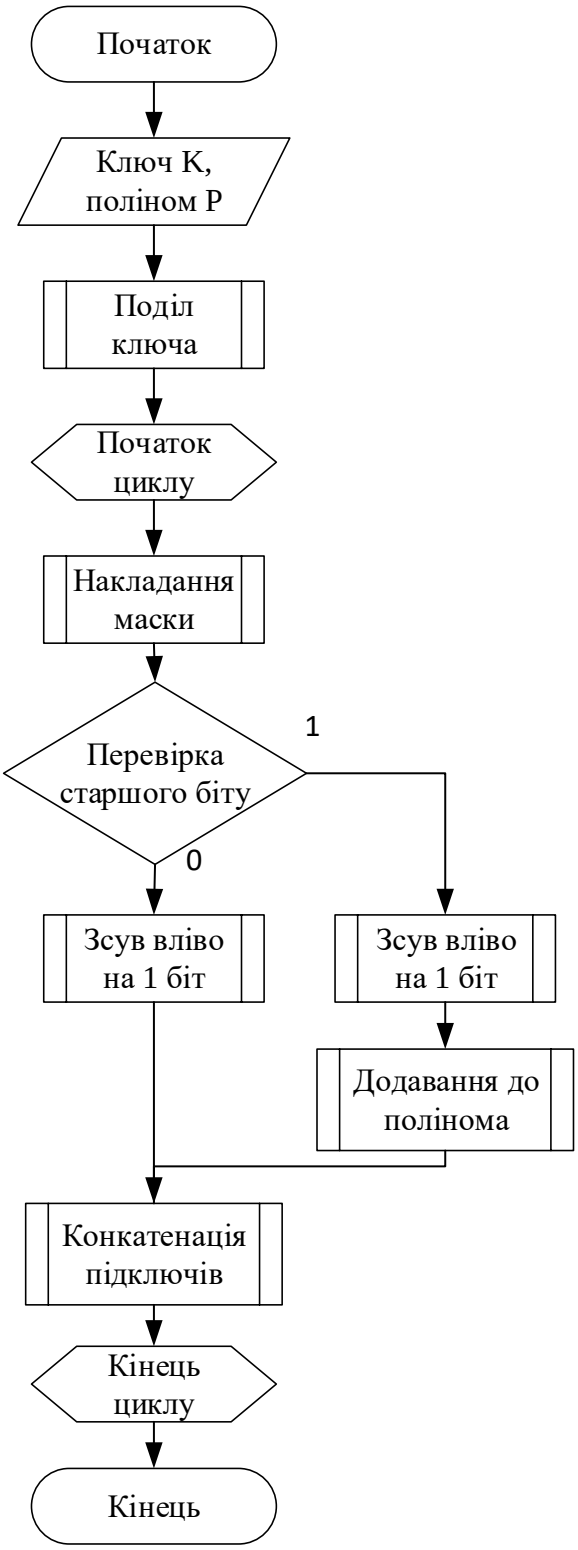
					08-20.МКР.012.00.000 І44			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Структурна схема псевдонедетермінованого блокового шифрування файлів	Літ.	Арк.	Аркушів
Перевір.		Баришев Ю.В.					1	1
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м		
Н. Контр.		Баришев Ю.В.						
Затверд.		Лужецький В.А.						

Інтерфейс засобу шифрування



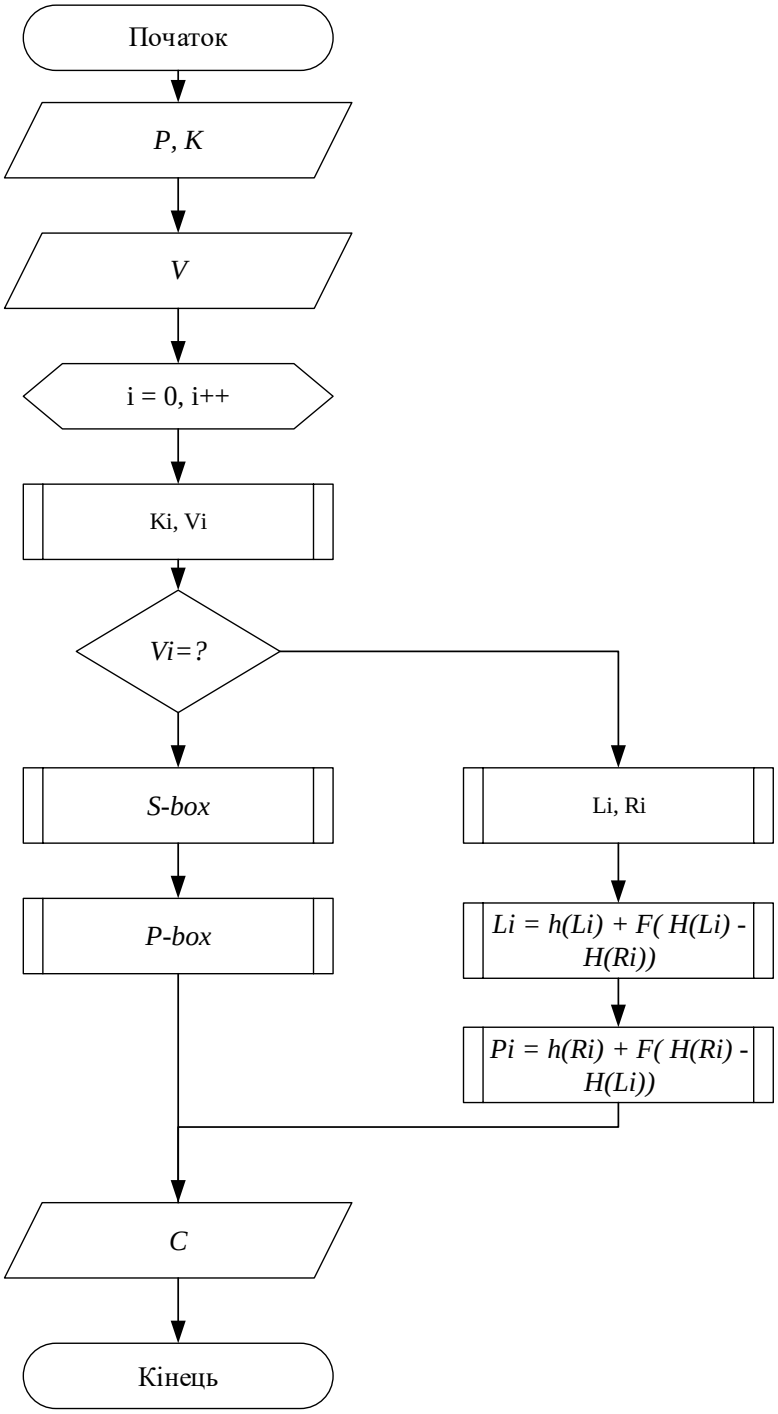
					08-20.МКР.012.00.000 І45			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Інтерфейс засобу шифрування	Літ.	Арк.	Аркушів
Перевір.		Баришев Ю.В.					1	1
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м		
Н. Контр.		Баришев Ю.В.						
Затверд.		Лужецький В.А.						

Алгоритм роботи блоку розгортання раундових ключів



					08-20.МКР.012.00.000 І46		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Алгоритм роботи блоку розгортання раундових ключів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Баришев Ю.В.					
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м	
Н. Контр.		Баришев Ю.В.					
Затверд.		Лужецький В.А.					

Алгоритм роботи засобу шифрування файлів (блок криптографічних перетворень)



					08-20.МКР.012.00.000 І47		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Ціхоцький М.С.			Метод та засіб псевдонедетермінованого блокового шифрування файлів. Алгоритм роботи засобу шифрування файлів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Баришев Ю.В.					
Рецензент		Савицька Л. А.				ВНТУ гр. 1 БС-20 м	
Н. Контр.		Баришев Ю.В.					
Затверд.		Лужецький В.А.					