

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ
СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

"На правах рукопису"
УДК _____

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“ ” _____ 2024 р.

Магістерська дисертація

на здобуття ступеня другого (магістерського) рівня вищої освіти
за освітньою програмою “Цифрові технології в енергетиці”
зі спеціальності 122 “Комп’ютерні науки”

на тему «Середовище випробування криптографічної стійкості методів
шифрування»

Виконав: студент 2 курсу, групи ТР-31мп

Вареніков Андрій Віталійович
(прізвище, ім’я, по батькові)

_____ (підпис)

Науковий керівник доцент, к.т.н., Світлана ШАПОВАЛОВА
(посада, науковий ступінь, вчене звання, ім’я ПРИЗВИЩЕ)

_____ (підпис)

Рецензент доцент кафедри ТАЕ, к.т.н.,

Артур РАЧИНСЬКИЙ

(посада, науковий ступінь, вчене звання, ім’я ПРИЗВИЩЕ)

_____ (підпис)

Н.контроль старший викладач, Ольга БЕСПАЛА

(посада, ім’я ПРИЗВИЩЕ)

_____ (підпис)

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних
посилань.

Студент _____
(підпис)

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра

ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – другий (магістерський)

спеціальність 122 “Комп’ютерні науки”

Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ

Завідувач кафедри ЦТЕ

Наталія АУШЕВА

«__»____ 2024 р.

ЗАВДАННЯ
на дипломну роботу студенту

ВАРЕНІКОВУ Андрію Віталійовичу

(прізвище, ім’я, по батькові)

1. Тема роботи **«Середовище випробування криптографічної**
стійкості методів шифрування»

керівник роботи **Шапалова Світлана Ігорівна, к.т.н., доцент**

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

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

2. Термін подання роботи студентом 06.12.2024 року

3. Вихідні дані до роботи середовище випробування криптографічної стійкості
методів шифрування

4. Перелік питань, які потрібно розробити _____

– проаналізувати сучасні методи шифрування та криптоаналізу;

– дослідити стійкість алгоритмів Цезаря, блокових методів, шифра Віженера, DES, AES, RC5 та Вернама до основних атак;

– розробити архітектуру середовища для тестування криптографічної стійкості методів шифрування;

– реалізувати програмне забезпечення для моделювання атак і перевірки алгоритмів

шифрування;

– провести апробацію програмного забезпечення та узагальнити результати тестування.

5. Орієнтований перелік ілюстративного матеріалу Блок-схеми алгоритмів шифрування, розшифрування та методів зламу, інтерфейс програми, структура взаємодії із середовищем.

6. Дата видачі завдання «15» вересня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	15.09.2023р.	
2	Аналіз вимог завдання, вибір методів і засобів розв’язання поставленої задачі	20.09.2023р.- 15.01.2024р.	
3	Підготовка матеріалів магістерської роботи	20.09.2023р.- 28.11.2024р.	
4	Підготовка публікацій	24.04.2024р.	
5	Захист програмного продукту	24.10.2024р.	
6	Передзахист	28.11.2024р.	

Студент

(підпис)

Андрій ВАРЕНІКОВ

(ім'я, ПРІЗВИЩЕ)

Керівник роботи

(підпис)

Світлана ШАПОВАЛОВА

(ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Актуальність теми. Зростання кількості інформаційних загроз і викликів до захисту даних вимагає вдосконалення методів криптографії. Розробка та аналіз алгоритмів шифрування, а також оцінка їх криптографічної стійкості є необхідними для забезпечення конфіденційності, цілісності та автентичності інформації в сучасному цифровому середовищі.

Метою роботи є створення програмного інструментарію для оцінювання стійкості базових алгоритмів шифрування.

Завдання дослідження:

- провести аналіз сучасних підходів криптографії;
- дослідити та порівняти базові алгоритми шифрування: метод Цезаря, блокові методи, DES, AES, RC5, шифр Віженера та метод Вернама;
- розробити програмне середовище для тестування та порівняння криптографічної стійкості базових алгоритмів шифрування;
- провести апробацію розробленого програмного забезпечення, визначити його ефективність.

Об’єкт дослідження – алгоритми шифрування та методи криптографії.

Предмет дослідження – криптографічна стійкість методів шифрування та їх уразливість до зламу.

Практична цінність отриманих у роботі результатів полягає в розробці програмного середовища для тестування криптографічної стійкості алгоритмів. Це дозволяє виявляти слабкі місця в шифрах і пропонувати рекомендації щодо їх удосконалення. Інструмент забезпечує зручність аналізу алгоритмів у наукових дослідженнях та практичному використанні.

Апробація результатів дисертації. Основні положення даної роботи доповідались та обговорювались на міжнародній науково-практичній конференції:

– Varenikov A.V., Shapovalova S.I. System of experimental research of cryptographic protection methods. *Сучасні проблеми наукового забезпечення енергетики*: матеріали XXI міжнар. наук.-практ. конф. молод. вчених і студентів у 2 т. 23–26 квіт. 2024 р. Київ. КПІ ім. Ігоря Сікорського, Вид-во «Політехніка». 2024. Т.2. С.224-225.

Дисертація складається з вступу, п'яти розділів та висновків. Повний обсяг дисертації складає 100 сторінок, в тому числі 91 сторінка основного тексту, 23 таблиці, 39 рисунків, 3 сторінки списку використаних джерел у кількості 50 найменувань.

Ключові слова: *криптографія, криптоаналіз, алгоритми шифрування, глибинне навчання, стійкість до зламу.*

ABSTRACT

Topic relevance. The growth of information threats and challenges to data protection requires the improvement of cryptographic methods. The development and analysis of encryption algorithms, as well as the assessment of their cryptographic stability are necessary to ensure the confidentiality, integrity and authenticity of information in the modern digital environment.

The purpose of the work is to study cryptanalysis methods, assess the stability of the main encryption algorithms and create software tools for testing them.

Research objectives:

- analyze modern approaches to cryptography and cryptanalysis;
- investigate and compare encryption algorithms such as Caesar, block methods, DES, AES, RC5, Vigenère cipher and Vernam method;
- develop a software system for testing the cryptographic stability of various encryption algorithms;
- conduct testing of the developed program, determine its effectiveness and functionality.

The object of the study is encryption algorithms and cryptanalysis methods.

The subject of the study is the cryptographic stability of encryption methods and their vulnerability to hacking.

The practical value of the results obtained in the work lies in the development of a software environment for testing the cryptographic stability of algorithms. This allows you to identify weaknesses in ciphers and offer recommendations for their improvement. The tool provides convenience for analyzing algorithms in scientific research and practical use.

Approbation of the results of the dissertation. The main provisions of this work were reported and discussed at:

– Varenikov A.V., Shapovalova S.I. System of experimental research of cryptographic protection methods. *Modern problems of scientific support of energy: materials of the XXI international scientific-practical conference of young scientists and students in 2 volumes.* April 23–26, 2024. Kyiv. Igor Sikorsky Kyiv Polytechnic Institute, Publishing house "Polytechnics". 2024. Vol. 2. P. 224-225.

The dissertation consists of an introduction, five chapters and conclusions. The full volume of the dissertation is 100 pages, including 91 pages of the main text, 23 tables, 39 figures, 3 pages of the list of sources used in the amount of 50 items.

Keywords: *cryptography, cryptanalysis, encryption algorithms, deep learning, resistance to hacking.*

ЗМІСТ

ВСТУП	9
1 ЗАДАЧА СТВОРЕННЯ СЕРЕДОВИЩА ВИПРОБУВАННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ МЕТОДІВ ШИФРУВАННЯ	12
1.1 Постановка задачі створення середовища випробування криптографічної стійкості методів шифрування.....	12
1.2 Методи шифрування для тестування криптографічної стійкості	14
1.2.1 Шифр Цезаря	14
1.2.2 Шифр Віженера.....	15
1.2.3 Блоковий шифр	16
1.2.4 Шифр DES	17
1.2.5 Шифр AES	19
1.2.6 Шифр Вернама	20
1.2.7 Шифр RC5	21
Висновки до розділу 1	22
2 ЗАСОБИ РОЗРОБКИ ТА РЕАЛІЗАЦІЯ МЕТОДІВ ШИФРУВАННЯ.....	24
2.1 Мова розмітки веб-сторінок HTML та стилі CSS.....	24
2.2 Мова програмування JavaScript.....	26
2.3 Реалізація метода Цезаря.....	27
2.4 Реалізація блокового методу.....	30
2.5 Реалізація методу Віженера	32
2.6 Реалізація методу DES.....	35
2.7 Реалізація методу AES.....	39
2.8 Реалізація методу RC5	42
2.9 Реалізація шифру Вернама.....	44

Висновки до розділу 2	46
3 РЕАЛІЗАЦІЯ МЕТОДІВ ЗЛАМУ	47
3.1 Метод перебору	47
3.2 Метод частотного аналізу	49
3.3 Метод пошуку ключа перебором	53
3.4 Атака методом відкритого тексту	54
3.5 Метод диференційного криптоаналізу	55
Висновки до розділу 3	58
4 РОБОТА КОРИСТУВАЧА ІЗ СЕРЕДОВИЩЕМ ВИПРОБУВАННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ МЕТОДІВ ШИФРУВАННЯ	59
4.1 Мінімальні вимоги системи для роботи із середовищем випробування криптографічної стійкості методів шифрування	59
4.2 Процес взаємодії із середовищем випробування криптографічної стійкості методів шифрування	60
4.3 Криптоаналіз методів шифрування	69
Висновки до розділу 4	72
5 РОЗРОБКА СТАРТАП ПРОЄКТУ	73
5.1 Опис ідеї стартап-проєкту	73
5.2 Технологічний аудит ідеї стартап-проєкту	75
5.3 Аналіз ринкових можливостей запуску стартап-проєкту	76
5.4 Розроблення ринкової стратегії стартап-проєкту	85
5.5 Розроблення маркетингової програми стартап-проєкту	89
ВИСНОВКИ	95
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	97
ДОДАТОК А	101

ВСТУП

На сьогоднішня захист інформації є однією із найважливіших задач людства. Люди дуже сильно покладаються на цифрові технології, а вони, нажаль, не є абсолютно надійними. Банківські рахунки, паролі від соціальних мереж, паролі від особистих кабінетів тощо. Злочинець зможе робити із викраденою інформацією будь-що: продати, використати для своїх потреб, шантаж тощо. Щоб не допустити цього, інформація має бути захищена надійними засобами. Особливо важливою проблема захисту інформації стає в нинішні часи – часи військового стану, коли будь-який витік інформації може привести до багатьох жертв та програшної ситуації на полі бою.

В світі існують такі засоби захисту інформації:

- технічний;
- організаційний;
- фізичний;
- правовий;
- програмний;
- криптографічний;
- адміністративний.

Всі види захисту інформації відрізняються за ступенем надійності та способом використання. Кожен із цих видів застосовується для окремих, конкретних випадків. Наразі ми розглянемо криптографічний захист, оскільки він забезпечує безпеку цифрової інформації, що зберігається на електронних носіях [1].

У сучасному світі, коли інформаційні технології стрімко розвиваються, зростають і способи незаконного здобуття інформації. Хоча рівень захисту постійно вдосконалюється, ризик перехоплення даних залишається високим через:

- складність алгоритмів зберігання даних;
- людський фактор (лінь, забудькуватість, необачність).

Зменшення ймовірності витоку інформації за допомогою криптографічного захисту є однією з найважливіших задач нашого часу.

Фактори, які збільшують ризик витоку інформації:

- інтенсивне зростання обсягу та кількості інформації, яка зберігається, розповсюджується й накопичується автоматизованими засобами;
- зберігання інформації різного призначення на тих самих носіях та у базах даних;
- надшвидке зростання кількості користувачів, які мають доступ до автоматизованих засобів і баз даних;
- автоматичний дозвіл доступу до баз даних та інформації з будь-якої точки світу;
- модернізація та вдосконалення обчислювальних алгоритмів, що застосовуються у комп'ютерних системах;

В усіх вище перерахованих випадках є ймовірність нелегального доступу до даних та їх редагування. Дуже велика кількість користувачів переймається тим, що їх дані можуть вкрати, тому їх захисту має приділятися висока увага. Ні для кого не секрет, що система зв'язку є дуже уразливою, а саме система зв'язку обчислювальних мереж. Зловмисники та злочинці мають можливість незаконно викрасти права на зберігання, копіювання, редагування та пошкодження інформації, що знаходиться в цих системах [2].

Наразі криптографічні методи використовуються не лише для захисту інформації від злочинців, що хочуть незаконним шляхом отримати до них доступ, а також грають надзвичайно важливу роль в таких сферах, як документообіг, анонімні голосування, грошові транзакції тощо.

Дана тема є актуальною через те, що на сьогоднішній день одним із найважливіших та найуразливіших ресурсів планети є інформація, котра

потребує ретельного захисту від осіб, що хочуть отримати несанкціонований доступ, незважаючи на заборону. Тема криптографічного захисту інформації є особливо актуальною саме в наші часи та в нашій країні, адже зараз іде війна, де будь-яка незахищена інформація, котра потрапить у ворожі руки, може призвести до значних втрат як серед військових, так і серед цивільних.

В умовах військового стану анонімність, засекреченість, конфіденційність та безпечність зв'язку стають надзвичайно важливими факторами для успішного виконання бойових задач. Методи криптографії повинні надавати захист військово важливої інформації від перехоплення та зламу шифру ворожою стороною. Використання найпередовіших алгоритмів криптографічного захисту та криптографічних систем значно збільшує надійність та недоступність обов'язкової мережі для ворога та інших недоброзичливців.

Отже, криптографія є не просто важливим, а необхідним фактором для успішної стратегії та успішного виконання бойових задач у нинішні часи.

1 ЗАДАЧА СТВОРЕННЯ СЕРЕДОВИЩА ВИПРОБУВАННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ МЕТОДІВ ШИФРУВАННЯ

Основна задача середовища Випробування криптографічної стійкості методів шифрування - це надання користувачу можливості шифрувати та дешифрувати введений ним текст, а також випробовувати обрані користувачем методи шифрування на стійкість до методів зламу [3]. Програма може використовуватися будь-ким, починаючи від дітей, котрі вирішили поглибитись у тему захисту інформації, закінчуючи справжніми військовими, котрим необхідно передавати один одному інформацію, про яку ворог ніколи не повинен дізнатися. Також програма може використовуватися банками, ІТ-компаніями, розробниками соціальних мереж та будь-якими сервісами, котрі потребують авторизації. В цьому розділі буде сформовано постановку задачі середовище випробування криптографічної стійкості методів шифрування та обґрунтовано вибір цих самих методів.

1.1 Постановка задачі створення середовища випробування криптографічної стійкості методів шифрування

Мета роботи – створити середовище, що дозволяє шифрувати та дешифрувати текст, а також визначати криптографічну стійкість обраних методів шифрування.

Для досягнення поставленої мети є необхідним виконання таких пунктів:

- визначитись із методами криптографії, що будуть реалізовані;

- розробити програмні алгоритми для визначених методів шифрування;

- розробити алгоритми дешифрування для криптографічних методів;
- розробити алгоритми зламу обраних методів;
- розробити середовище за поставленою метою;
- протестувати розроблене середовище.

Середовище тестування повинне надавати користувачу такий перелік можливостей:

- вводити текст з клавіатури на декількох мовах для подальшого шифрування та дешифрування;

- обирати метод шифрування із списку;
- шифрувати текст відповідним криптографічним методом;
- дешифрувати текст відповідним криптографічним методом;
- вводити криптографічний ключ з клавіатури на декількох мовах за необхідністю;

- тестувати методи шифрування на криптографічну стійкість;
- виводити результати шифрування;
- виводити результати дешифрування;
- виводити результати тестування.

Середовище тестування криптографічної стійкості методів шифрування може використовуватись як в цивільних цілях, так і в військових, адже надійний захист інформації є важливою частиною власної безпеки.

1.2 Методи шифрування для тестування криптографічної стійкості

Криптографічні засоби захисту — це методи і засоби, задача котрих — захистити інформацію від несанкціонованого доступу, унеможливаючи прочитання цієї інформації для людей, котрим заборонено доступ.

Криптографія зазвичай використовується для передачі інформації на відстані через певні канали зв'язку, а також для зберігання інформації в захищеному від сторонніх поглядів вигляді [4]. Методи шифрування працюють за двома принципами:

- перестановка літер, при якій обсяг та кількість літер у тексті залишаються незмінними;
- стиснення чи розширення вхідної інформації при використанні перетворень за допомогою ключів.

1.2.1 Шифр Цезаря

Перший метод шифрування, що необхідний для виконання задачі тестування криптографічної стійкості, є шифр Цезаря. Шифр Цезаря — це історичний метод шифрування, який використовувався ще в античні часи за правління Юлія Цезаря, а власне він і винайшов цей метод.

Згідно з історичними свідченнями, цей метод використовувався для передачі військових повідомлень. Довідка вказує, що метод Цезаря працював шляхом зміщення літер латинського алфавіту на декілька позицій. Сам Юлій Цезар використовував три позиції, щоб зробити текст нечитабельним. Принцип роботи шифру дуже простий: кожна літера вхідного тексту замінюється на іншу,

яка знаходиться на певному діапазоні в алфавіті. Наприклад, якщо ключ зсуву дорівнює три, то літера «А» стає «Г», «Б» — «Д» і так далі [5].

У роки правління Юлія Цезаря цей метод шифрування вважався досить надійним, але це було через те, що ніхто не знав, як він працює. Проте сьогодні шифр Цезаря не є актуальним і надійним, оскільки його легко зламати, наприклад, звичайним перебором кількості позицій зсуву. Обмежена кількість варіантів зсуву робить його дуже уразливим до методу грубої сили.

Попри свою простоту, шифр Цезаря відіграв важливу роль у розвитку криптографії, породивши багато інших криптографічних методів, зокрема шифр Віженера, який також використовується в сучасних додатках. Незважаючи на свою примітивність, цей метод шифрування має велике історичне значення і був одним із перших прикладів систематичного підходу до шифрування повідомлень.

Незважаючи на свій вік, шифр Цезаря може використовуватися і в сучасних реаліях: він часто застосовується для навчальних цілей, розважальних задач, а також у безпечних додатках для захисту паролів і в історичних дослідженнях.

1.2.2 Шифр Віженера

Наступним методом шифрування, необхідним для виконання поставленої задачі, є шифр Віженера. Цей метод був винайдений Блезом де Віженером, французьким дипломатом і криптографом, в 1586 році [6].

В свій час це був найпроривніший метод криптографічного захисту. По суті, це вдосконалений метод шифра Цезаря, бо він також використовує зсуви за певним значенням. Але на відміну від шифра Цезаря, шифр Віженера значно надійніший, бо використовує не один зсув для всіх літер, а за допомогою ключового слова визначається, на скільки позицій треба змінити відповідну літеру.

Наприклад, якщо ключове слово — «крипто», то перша літера шифрованого тексту буде визначатися першою буквою ключа, тобто літерою «к», для другої літери зсув буде визначений за допомогою другої літери ключа, тобто літери «р», для третьої — за допомогою літери «и», і так далі. Якщо ключове слово коротше за текст, воно повторюється. Цей метод значно надійніший за шифр Цезаря, оскільки кожна літера тексту може бути зашифрована різними зсувами, що значно підвищує стійкість шифру до частотного аналізу.

Однак цей шифр не є абсолютно незламним. На жаль, він застарів, і для його зламу можна використовувати методи аналізу або вже існуючі таблиці Віженера для пошуку закономірностей у зашифрованому тексті. Але на свій час цей шифр був одним із найнадійніших. Якщо він був найнадійнішим у своєму часі, то в сучасному світі він вже не використовується для серйозного захисту важливої інформації. Проте він все ще може служити людству в сучасному світі, оскільки використовується для навчальних цілей, розважальних цілей, історичних та культурно-освітніх проектів [7].

З цього можна зробити висновок, що шифр Віженера застарів для захисту інформації, проте й досі знаходить своє місце в певних сферах застосування.

1.2.3 Блоковий шифр

Наступним методом є простий блоковий шифр на прикладі методу перестановки. Метод перестановки є одним із найпростіших блокових шифрів, які працюють за допомогою перестановки символів або байтів поблочно та за певним алгоритмом. На відміну від шифрів заміни, шифр перестановки зберігає символи в їх оригінальному вигляді, але змінює їхнє місце в шифрованому повідомленні. За допомогою цих дій створюється так званий ефект перемішування інформації, що робить її нечитабельною для інших осіб.

Метод працює таким чином, що повідомлення розбивається на блоки певної довжини, яка має бути однаковою, наприклад, по 8 або 16 символів. Після цього кожен блок піддається перестановці його літер або символів за допомогою певного ключа, який вказує, як треба розмістити символи в зашифрованому варіанті тексту. Цей метод досить простий і його легко реалізувати, але його безпека досить низька за нинішніми стандартами. Основна його проблема — це можливість, як і у методів Цезаря та Віженера, бути легко зламаним методом грубої сили: зловмисник буде перебирати всі можливі варіанти, але вони є досить обмеженими через невеликі розміри блоків, тому він дуже швидко його підбере при нинішньому рівні технологій.

Крім того, шифр уражається частотним аналізом, бо структура шифру залишається такою ж, як і у вхідного тексту, змінюється лише порядок літер. Як і попередні методи, він застарів, проте відіграв досить значну роль в історії сучасної криптографії. На його основі були побудовані сучасні методи, такі як DES і AES, де застосовуються більш складні схеми перестановки. У сучасному світі його використання досить обмежене, але, як і попередні методи, він використовується в навчальних цілях для розуміння основних принципів шифрування. Крім того, дані, які не потребують високого рівня захисту, можуть бути захищені цим методом криптографії.

1.2.4 Шифр DES

Ще один метод, необхідний для реалізації постановки задачі, — це шифр DES (Data Encryption Standard). Цей метод є одним із найвідоміших симетричних шифрів, який був розроблений компанією IBM. Шифр DES використовувався як національний стандарт для шифрування даних в США з 1977 року [8]. Метод працює з блоками даних розміром 64 біти і використовує ключ довжиною 56 біт

для шифрування тексту. Алгоритм використовувався в багатьох сферах життя, таких як банки, електронні платежі, передача конфіденційної інформації тощо. Активно використовувався до 90-х років, проте був замінений більш надійними аналогами.

Криптографічний метод шифрування DES використовує 16 раундів заміни і перестановок, які включають в себе математичні операції XOR. Під час перетворення тексту в шифротекст кожен біт піддається перетворенню, що робить текст абсолютно нечитабельним і досить складним для зламу без знання відповідного ключа. Цей шифр був розроблений для використання на апаратному рівні, що було дуже популярним у ті часи. Однак з часом DES став ненадійним через недостатню довжину ключа, яка була неадекватною для забезпечення криптографічної стійкості, оскільки комп'ютерні технології розвивалися швидкими темпами. Потужні комп'ютери дозволяли зламати цей шифр, застосовуючи методи грубої сили, перебираючи всі можливі варіанти ключів.

Через це в середині 2000-х років метод шифрування DES був замінений більш надійним методом під назвою AES, про який буде розказано далі. Метод DES безумовно застарів, проте на свій час він був одним з найнадійніших. Принципи шифрування DES використовуються для розробки новітніх алгоритмів шифрування, які застосовуються й сьогодні. На сьогодні шифр DES все ще використовується в деяких застарілих системах, де його застосування неможливо замінити на новіші методи через високу вартість оновлення. Ці системи були захищені методом DES у той час, коли вони розроблялись, хоча новітнє обладнання вже використовує більш надійні методи шифрування.

1.2.5 Шифр AES

Наступним методом є метод AES (Advanced Encryption Standard). Це покращений метод DES і один із найпоширеніших та найбезпечніших сучасних методів симетричного шифрування. Цей метод криптографічного захисту був розроблений для заміни застарілого шифру DES і був обраний стандартом криптографічного захисту США в 2001 році. Метод розробив американець Джоан Даміан.

AES використовує блочне шифрування з розміром блоку 128 біт, а довжина ключа може варіюватися від 128 біт до 256 біт. Найбільша перевага цього методу — його криптографічна стійкість та висока ефективність. Як і метод DES, він працює за принципом множинних раундів перетворень, проте кількість раундів залежить від довжини ключа: чим довший ключ, тим більше перетворень. На відміну від DES, ключ AES набагато довший, що гарантує високий рівень безпеки і майже неможливість зламати шифр [9].

Криптографічний метод AES є симетричним шифром, що означає, що однаковий ключ використовується як для шифрування, так і для дешифрування даних. Це робить його зручним у використанні, але створює високі вимоги до безпеки ключа. Якщо ключ буде занадто простим, шифр легко зламається. Основний напрямок використання цього методу — захист конфіденційної інформації в урядових і комерційних системах. Наприклад, для захисту даних на дисках, в базах даних, в системах передачі даних тощо. Також його використовують у системах мобільного зв'язку та фінансових додатках.

На сьогодні метод AES є потужним рішенням у світі криптографічного захисту і використовується в багатьох сферах — від фінансових установ до військових систем. Хорошим прикладом є VPN додатки, які використовують цей метод для захисту передачі даних в інтернеті [10]. Також цей метод є досить

швидкодіє, і саме завдяки швидкості і стійкості він залишається актуальним у сучасному світі для захисту від технологічних загроз.

1.2.6 Шифр Вернама

Далі на черзі шифр Вернама, також відомий як одноразовий блокнот. Цей метод криптографії є унікальним, тому що прийнято вважати, що він є ідеальним з точки зору безпеки. Шифр був розроблений Гілбертом Вернамом у 1917 році і працює за рахунок використання ключа, довжина якого повинна дорівнювати довжині повідомлення, що буде зашифроване.

Суть методу полягає в тому, що кожен біт або символ тексту, що буде шифруватися, замінюється відповідним бітом або символом за допомогою математичної операції XOR. Таке повідомлення неможливо зламати або дешифрувати без знання ключа.

Абсолютна безпека залежить від декількох умов. По-перше, ключ має бути абсолютно випадковим. По-друге, ключ має використовуватися тільки один раз. По-третє, ключ не повинен розповсюджуватися. Якщо виконати всі ці умови, такий шифр неможливо зламати будь-яким методом. Тобто, навіть якщо зломисник матиме безкінечні обчислювальні ресурси, він все одно не зможе досягти задуманого.

Але, незважаючи на абсолютну криптографічну стійкість, цей шифр має певні недоліки. Головна проблема полягає в генерації і зберіганні випадкових ключів, довжина яких дорівнює довжині повідомлення. Забезпечити унікальність та унеможливити повторне використання ключа є досить складною задачею. Тому шифр Вернама не дуже підходить для захисту великих обсягів інформації, навіть якщо потрібна максимальна безпека.

Прикладом використання цього методу є Друга світова війна, коли інформація на полі бою та між союзниками шифрувалася саме цим методом. У сучасних реаліях метод іноді використовується для передачі надзвичайно важливих даних, таких як дипломатичні повідомлення або військові вказівки. Шифр Вернама також використовується в навчанні для демонстрації принципів роботи ідеального криптографічного захисту.

Цей шифр відіграє велику роль в історії криптографії, тому що показав, що абсолютна криптографічна стійкість можлива, хоча й важка в реалізації на практиці.

1.2.7 Шифр RC5

Останнім шифром для реалізації поставленої задачі є метод RC5. RC5 — це симетричний блоковий шифр, розроблений Рональдом Рівестом у 1994 році.

Особливістю цього шифру є простота реалізації та висока гнучкість. Він підтримує різні розміри блоку: 32, 64 та 128 біт, а довжина ключа може варіюватися від 0 до 2040 біт [11]. Кількість раундів шифрування обирається в діапазоні від 0 до 255. Завдяки такій гнучкості цей криптографічний метод можна налаштувати для різних рівнів безпеки.

За основу роботи RC5 взяті математичні операції: додавання, XOR та циклічний зсув. Ці операції досить ефективні для реалізації як у програмному забезпеченні, так і в апаратних засобах. Під час шифрування текст розбивається на блоки, і кожен із цих блоків піддається перетворенням залежно від введеного ключа. Ключ у цьому методі проходить етап попередньої обробки, під час якої генерується таблиця підключів, що використовується на кожному раунді. Підключі створюються за допомогою рекурсивного додавання та циклічного зсуву, що забезпечує високу криптографічну стійкість.

Однією з особливостей RC5 є те, що його криптографічна стійкість безпосередньо залежить від кількості раундів. Чим більше раундів використовується, тим вищий рівень безпеки, і навпаки — менша кількість раундів знижує стійкість. Проте при використанні достатньої кількості раундів RC5 забезпечує високий рівень захисту. Його простота є важливою перевагою для використання на слабких комп'ютерах.

Цей метод широко застосовувався в минулому для захисту даних у банківських системах, мережевих протоколах та системах зберігання інформації. Сьогодні RC5 використовується рідше через появу більш сучасних і потужних алгоритмів, таких як AES. Проте RC5 усе ще застосовують у випадках, де потрібна висока продуктивність і гнучкість налаштувань, а також у системах зі слабкими апаратними засобами.

В історії криптографії RC5 відіграв значну роль. Його простота та гнучкість зробили його важливим об'єктом для досліджень та навчання у сфері криптографії.

Висновки до розділу 1

Підводячи підсумки до цього розділу, можна сказати, що розробка середовища тестування криптографічної стійкості методів шифрування є важливим та потужним рішенням у створенні інструменту для оцінки безпеки та ефективності алгоритмів шифрування. Поставлене завдання включає в себе розробку програмного забезпечення, що надає користувачеві можливості шифрування, дешифрування тексту та аналізу стійкості методів шифрування до різноманітних атак. Також система надає можливість протестувати методи на практиці.

Для досягнення поставленої мети необхідно реалізувати такі методи шифрування: шифр Цезаря, шифр Віженера, блочний шифр, шифр DES, шифр AES, шифр Вернама та шифр RC5. Кожен із цих методів відрізняється за своїм призначенням та криптографічною стійкістю [12]. Шифр Цезаря та шифр Віженера демонструють принципи роботи історичних алгоритмів шифрування, які відіграли значну роль у розвитку сучасної криптографії. Блочний шифр демонструє основу шифрування через перестановки та розбиття тексту на блоки для подальшого розуміння більш складних алгоритмів шифрування. Шифри DES та AES є гарними прикладами більш сучасних методів захисту даних, починаючи від перших спроб стандартизації у темі захисту даних і закінчуючи більш надійним і сучасним методом, який використовується і в повсякденному житті, і в нинішні часи.

Шифр Вернама показує приклад ідеального методу шифрування, який має абсолютну криптографічну стійкість і унеможливорює процес його зламу. Шифр RC5 демонструє приклад гнучкості реалізації та гарної надійності в сучасних умовах. Середовище тестування криптографічної стійкості методів шифрування має широкий спектр застосувань, починаючи від навчальних цілей і закінчуючи військовими. Реалізація поставленого завдання забезпечить не лише практичне використання, а й гарну основу для подальших досліджень методів криптографії.

2 ЗАСОБИ РОЗРОБКИ ТА РЕАЛІЗАЦІЯ МЕТОДІВ ШИФРУВАННЯ

Мовою програмування для реалізації середовища з випробування криптографічної стійкості методів шифрування було обрано JavaScript. Це все через те, що дана мова програмування створювалася для браузерів, а це означає, що застосунок буде кросплатформним та доступним на всіх операційних системах, таких як iOS, Android, Windows, Linux, macOS тощо.

Також JavaScript має велику кількість бібліотек, серед яких є бібліотеки для криптографії. Якщо їх використовувати, то реалізація середовища стане значно легшою. Інтерфейс програми побудовано мовою розмітки вебсторінок HTML, а також були використані стилі CSS. Програмний код був написаний у середовищах Visual Studio та Notepad++ [13].

Ці середовища розробки значно полегшують процес написання коду, тому що в них є інтерактивна підсвітка помилок, тегів та функцій. До того ж Notepad++ був розроблений саме для побудови вебпрограм.

В цьому розділі будуть описані методи реалізації, що були застосовані для написання коду програми, а також кроки, які необхідно виконати для роботи методів шифрування.

2.1 Мова розмітки веб-сторінок HTML та стилі CSS

HTML (HyperText Markup Language) є основною мовою для створення інтерфейсу будь-яких вебзастосунків. Завдяки цій мові розробникам можна досить легко реалізовувати інтуїтивно зрозумілий дизайн для своїх програм, як і

в нашому випадку — для реалізації середовища тестування криптографічної стійкості методів шифрування. HTML дозволяє досить зручно структурувати контент та реалізовувати взаємодію користувача з інтерфейсом за допомогою таких засобів, як форми для введення даних, поля для введення тексту, відображення та вивід інформації на екран тощо. Якщо використовувати HTML для реалізації програмного продукту, то користувачеві буде досить легко та інтуїтивно працювати з розробленими алгоритмами шифрування та зламу.

HTML підтримує інтеграцію з JavaScript, а це означає, що розробнику стають доступними потужні бібліотеки, функції та математичні обчислення, що дозволить більш точно та ефективно розробити необхідні пункти для виконання поставленої задачі. Також HTML сумісний із CSS та фреймворками, такими як Bootstrap. За допомогою них можна зробити адаптивний інтерфейс, котрий буде автоматично підлаштовуватися під будь-який екран — як настільні комп'ютери, так і смартфони.

У випадку тестування методів криптографічної стійкості адаптивність є досить важливою, тому що вона дозволяє дослідникам і розробникам працювати із середовищем у будь-яких умовах та на будь-яких пристроях.

Враховуючи всі вищеперелічені фактори, можна зробити висновок, що HTML є гарним вибором для реалізації інтерфейсу середовища тестування криптографічної стійкості методів шифрування. Таким чином, HTML є оптимальним вибором для реалізації середовища тестування криптографічної стійкості. Його простота, гнучкість і сумісність із сучасними веб-технологіями дозволяють створити ефективний інструмент, який відповідатиме науковим і практичним вимогам магістерської роботи.

CSS (Cascading Style Sheets) відповідає за стиль і оформлення сторінок в середовищі тестування. У даній роботі CSS дозволяє покращити дизайн веб-сторінок та створити естетичний та привабливий дизайн, що є досить важливим для успішної та комфортної взаємодії користувача із системою. За допомогою

CSS структурувати елементи веб-сторінки стає значно легше, тому що він дозволяє виділяти ключові елементи, розташовувати їх в правильній послідовності та красиво оформляти [14].

Як вже згадувалося при описі HTML, адаптивність є досить важливим фактором в реалізації системи. І саме завдяки CSS адаптивність можна досягнути. Завдяки правильним налаштуванням стилів веб-сторінка може виглядати привабливо на будь-яких пристроях з різною діагоналлю екрана.

Також стилі CSS дозволять додавати інтерактивні елементи, що може зробити взаємодію користувача із системою більш зручною та зрозумілою. Наприклад, виділення та підсвітка активних елементів або динамічне відображення результатів шифрування, дешифрування та тестування.

2.2 Мова програмування JavaScript

У середовищі тестування криптографічної стійкості методів шифрування всі методи, як шифрування, так і зламу, написані за допомогою мови програмування JavaScript. JavaScript є мовою програмування, яка використовується виключно для вебдодатків. Ця мова програмування є високорівневою та дозволяє створювати вебсторінки, котрі мають інтерактивний інтерфейс для взаємодії з користувачем. До того ж, JavaScript може працювати в клієнт-серверній архітектурі як на стороні сервера, так і на стороні клієнта. Це реалізується за допомогою платформи Node.js [15].

JavaScript має багато вбудованих функцій та велику кількість бібліотек, що значно спрощує реалізацію середовища тестування та розширює його можливості. У JavaScript є багато переваг перед іншими мовами програмування. Наприклад, мова є інтерпретованою, тобто код може виконуватися миттєво, без попередньої компіляції з використанням компіляторів або інших програм. Також

JavaScript, як вже було сказано раніше, є мовою для розробки вебдодатків, тому всі програми, написані цією мовою, є кросплатформенними.

До того ж, у JavaScript є необхідні для реалізації методів шифрування бібліотеки, що значно спрощує розробку цих методів та покращує їх надійність і швидкодію. Останньою перевагою є те, що мова програмування JavaScript має відкритий код, і це дозволяє розробникам самостійно вдосконалювати її, створювати власні бібліотеки та розвивати мову спільними зусиллями. JavaScript є досить безпечною мовою, адже браузер, у яких вона використовується, не потребує спеціальних доступів до внутрішньої файлової системи, яка може бути ураженою вірусами та шкідливими програмами.

2.3 Реалізація метода Цезаря

Шифрування методом Цезаря ґрунтується на ідеї циклічного зміщення літер алфавіту на фіксовану кількість позицій, котра задається з клавіатури.

Уявімо, що алфавіт є замкненим списком, де після останньої літери знову йде перша. Кожна літера змінюється за заданим зсувом, але символи, які не належать до алфавіту (пробіли, розділові знаки, цифри тощо), залишаються без змін [16].

Процес шифрування виконується за наступними кроками:

- Обробка тексту. Текст розбивається на окремі символи, щоб з кожною літерою можна було працювати окремо. Це робиться для того, щоб шифрування стосувалося лише літер і дозволяло визначити кожен їх позицію в алфавіті.
- Розпізнавання символів. Коли текст розбито, для кожної літери визначається її номер в алфавіті залежно від мови. Наприклад, у латинському

алфавіті літерам від А до Z присвоюються номери від 0 до 25. Для малих літер діапазон такий самий, але враховується їх регістр.

- Додавання зсуву. Коли позиції визначено, до них додається число зсуву, введене користувачем. Наприклад, якщо початкова літера С і зсув дорівнює трьом, то $2 + 3 = 5$, тобто літера С перетворюється на літеру F. Якщо результат перевищує діапазон алфавіту, список літер продовжується циклічно. Наприклад, для літери Z зі зсувом 3 буде визначено позицію 28, але, оскільки латинський алфавіт має лише 26 літер, результатом стає цифра 2, тобто літера С.

- Збереження символів, що не є літерами. Символи, які не є частиною алфавіту (наприклад, цифри, пробіли або розділові знаки), не шифруються і залишаються незмінними на своїх місцях.

- Повернення результату. Усі перетворені символи та незмінні символи збираються в один рядок, який і є зашифрованим текстом.

Отже, виконуючи всі ці кроки, отримуємо зашифрований текст.

Алгоритм шифрування метода Цезаря зображено на схемі (Рисунок 2.1).



Рисунок 2.1 – Блок-схема алгоритму шифрування метода Цезаря

Процес розшифрування майже ідентичний до процесу шифрування, але є його оберненою версією. Кожна літера повертається назад на ту кількість позицій, яка використовувалася для шифрування. Якщо шифрування виконувалося вперед, то розшифрування виконується у зворотному напрямку.

Процес розшифрування виконується за наступними кроками:

- Обробка зашифрованого тексту. Як і у випадку шифрування, текст розбивається на окремі символи, щоб працювати з кожною літерою окремо.
- Визначення початкової позиції літер. Кожній літері присвоюється номер в алфавіті. Якщо під час шифрування до позиції додавалося значення зсуву, то під час розшифрування воно віднімається.
- Віднімання зсуву. Із позиції кожної літери віднімається значення, введене з клавіатури. Якщо результат менший за нуль, використовується та сама циклічність, що й у процесі шифрування.
- Збереження незмінних символів. Як і під час шифрування, символи, що не належать алфавіту, залишаються незмінними.
- Збирання тексту. Після обробки літер і додавання незмінних знаків текст збирається в цілий рядок, який є розшифрованим текстом.

Розшифрування забезпечує повне відновлення вихідного тексту за умови правильного застосування зсуву. Реалізація цього процесу в програмному середовищі дозволяє не тільки перевірити коректність алгоритму шифрування, а й продемонструвати практичну зручність використання метода, що є важливим для аналізу його ефективності та надійності.

На схемі нижче продемонстровано алгоритм роботи розшифрування методом Цезаря (Рисунок 2.2).



Рисунок 2.2 – Блок-схема алгоритму розшифрування методом Цезаря

Виходячи з даної схеми, можна побачити, що процес шифрування майже ідентичний процесу розшифрування.

2.4 Реалізація блокового методу

Шифрування блоковим методом чимось схоже на створення нового секретного коду, у якому кожна літера має свій аналог в обличчі іншої літери. Ідея шифрування блоковим методом полягає в тому, щоб перемішати алфавіт у випадковому порядку, створити ключ шифрування і використати цей ключ для заміни символів тексту, введеного з клавіатури.

Якщо казати простіше, то потрібно взяти алфавіт, перемішати його і потім використати цей перемішаний алфавіт як карту, котра вказує, на яку літеру потрібно замінити кожен символ в тексті [17].

Для того щоб зашифрувати текст блоковим шифром, необхідно виконати певні кроки:

- Підготовка тексту. Спочатку введений текст приводять до робочого вигляду: у даному випадку всі літери змінюють свій регістр на заглавний, щоб не виникало плутанини з великими та малими літерами.
- Робота з кожним символом. Далі кожна літера введеного тексту проходить перевірку: Якщо літера є частиною алфавіту, їй присвоюється номер у цьому самому алфавіті. Потім у ключі на цій позиції за номером шукається літера, відповідна цьому номеру. Наприклад, якщо на першій позиції стоїть літера А, а на першій позиції в ключі стоїть літера С, то А стає літерою С.
- Збереження незмінних символів. Пробіли, розділові знаки тощо не будуть перероблені алгоритмом шифрування.
- Формування зашифрованого тексту. Після того як ці символи оброблені шифрувальним алгоритмом, до них додаються символи, що не пройшли змін, і об'єднуються в один єдиний зашифрований текст.

На схемі нижче продемонстровано схематично роботу алгоритму шифрування (Рисунок 2.3).

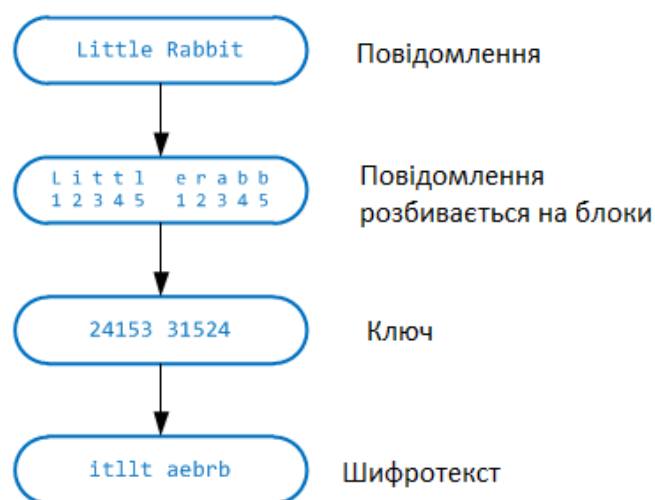


Рисунок 2.3 – Блок-схема алгоритму блокового методу

Після шифрування створюється ілюзія хаосу: текст є абсолютно нечитабельним і виглядає як якась нісенітниця. Проте кожна літера має чітку відповідність у ключі.

Розшифрування - це обернений процес до шифрування. Ключ має бути тим самим, але відповідність шукається у зворотному напрямку: замість того, щоб замінювати літери алфавіту на ключові, ми будемо змінювати ключові літери на літери алфавіту.

Ключ, який використовувався для шифрування, зараз стає дешифратором. Щоб виконати розшифрування, необхідно пройти такі кроки:

- Підготовка тексту. Зашифрований текст, як і під час шифрування, розбивається на символи для обробки. Також він повністю переводиться у верхній регістр, тобто всі літери стають великими.
- Робота з кожним символом. Для кожної літери тексту перевіряється її позиція в ключі: Якщо на першому місці ключа стоїть літера С, а в алфавіті літера А, то С стає А. Збереження незмінних символів. Як і під час шифрування, символи, що не є літерами (розділові знаки, пробіли тощо), залишаються без змін.
- Формування вихідного тексту. Коли всі символи пройшли обробку алгоритмом, вони поєднуються із символами, що не проходили обробку, і збираються в один уже читабельний текст.

Після процедури розшифрування текст знов стає читабельним.

2.5 Реалізація методу Віженера

Наступним криптографічним методом є метод Віженера. Він є наступним за рівнем методом шифрування. Суть його в тому, що кожна літера тексту шифрується за допомогою ключа, який складається з декількох символів. Це

створює ефект так званого ковзаючого шифру, де кожна літера зміщується по алфавіту на різну кількість позицій, визначену відповідно літерою ключа.

Основна ідея заключається в тому, що кожна літера тексту отримує свій зсув за допомогою ключа. Ключ багаторазово повторюється до того моменту, поки його довжина не стане дорівнювати довжині вхідного тексту [18]. Алфавіт використовується циклічно, де після останньої літери йде перша.

Щоб зашифрувати текст методом Віженера, потрібно виконати такі кроки:

- Підготовка тексту та ключа: Текст та ключ переводяться в один регістр, а саме в верхній регістр, щоб не було плутанини між маленькими та великими літерами. Якщо ключ коротший за текст, він буде повторюватися стільки разів, скільки потрібно, поки його довжина не вирівняється з довжиною вхідного тексту. Наприклад, для тексту "Hello" з ключем "Key", ключ розширюється до "KEYKE".

- Застосування зсуву: Кожна літера порівнюється з літерою ключа. Наприклад, перша літера тексту "Hello" — Н. Вона шифрується літерою ключа "K". Літера "K" в алфавіті — це 11-та літера, тому Н зміщується на 10 позицій уперед, стаючи R. Зсув для кожної літери визначається її порядковим номером у алфавіті. Наприклад, літера "A" — це 0, "B" — 1, "C" — 2, "Z" — 25.

- Обробка символів, які не належать до алфавіту: Символи, які не є літерами, тобто пробіли, розділові знаки тощо, залишаються без змін. Це робиться для того, щоб шифрування не порушило структуру тексту.

- Створення зашифрованого тексту: Після того як текст було оброблено, він поєднується із розділовими знаками та пробілами і з'єднується в один нечитаємий текст.

Алгоритм роботи методу шифрування Віженера зображено на схемі (Рисунок 2.4), яка ілюструє послідовність дій для процесу шифрування тексту. Метод ґрунтується на використанні ключового слова, яке повторюється до відповідної довжини відкритого тексту. Кожна літера відкритого тексту

шифрується шляхом додавання її значення (згідно з алфавітом) до значення відповідної літери ключа, обчисленого за модулем кількості літер у алфавіті. У результаті формується зашифрований текст, який значно стійкіший до частотного аналізу, ніж простіші методи, наприклад, шифр Цезаря.

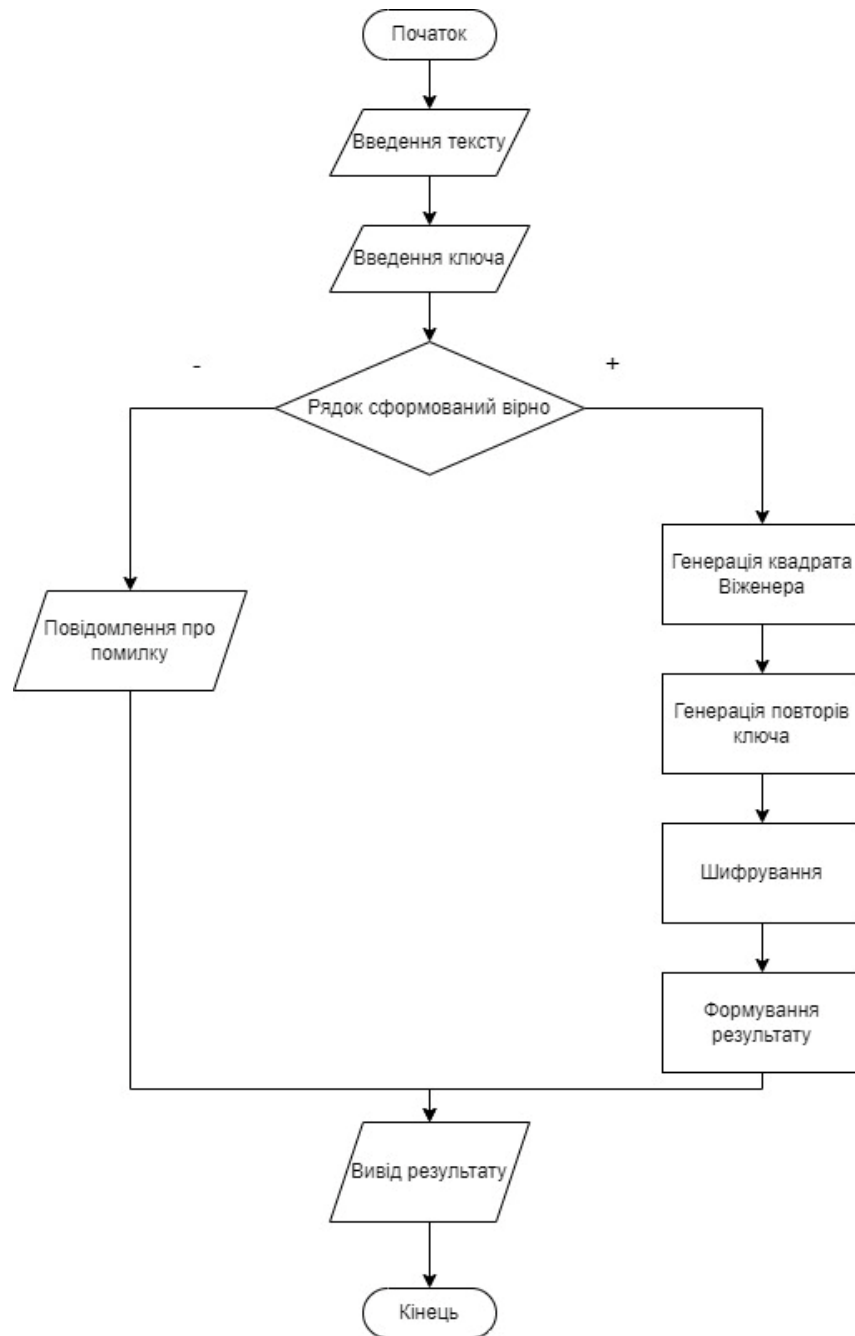


Рисунок 2.4 – Блок-схема алгоритму шифрування методом Віженера

Щоб розшифрувати шифр Віженера, необхідно виконати обернений процес шифрування. Він використовує той самий ключ, але замість додавання віднімає його. Це дозволяє повернути кожен перетворену літеру до її початкового стану. Щоб виконати дешифрацію шифру Віженера, необхідно пройти такі кроки:

- Підготовка тексту та ключа: Зашифрований текст і ключ готуються так само, як і під час процесу шифрування. Текст і ключ приводяться до великого регістру, а ключ зациклюється до того, поки його довжина не порівняється з довжиною вхідного тексту.
- Зворотний зсув: Для кожної літери зашифрованого тексту визначається її позиція в алфавіті, а потім віднімається зсув, що буде відповідати літері ключа. Наприклад, якщо зашифрована літера "R", а літера ключа — "K", то позиція "R" — 18. Віднімаємо 10, і отримуємо "H".
- Збереження незмінних символів: Як і під час шифрування, символи, що не є літерами (розділові знаки, пробіли тощо), залишаються на місці.
- Створення розшифрованого тексту: Оброблені літери поєднуються із незмінними символами, і відтворюється початковий незашифрований текст. Після процедури розшифрування текст знову стає читабельним.

2.6 Реалізація методу DES

Метод шифрування DES (Data Encryption Standard) можете реалізувати за допомогою використання бібліотеки мови JavaScript Crypto.js. Crypto.js є гарним вибором для реалізації цього методу шифрування з декількох причин.

Перш за все, ця бібліотека дуже зручна для розробників через те, що дуже легко використовується її API, що дозволяє її без будь-яких проблем інтегрувати в свою програму [19]. Також Crypto.js можна використовувати як на серверній стороні, так і на клієнтській. Більш того, ця бібліотека підтримує ряд популярних

та потужних криптографічних алгоритмів, серед яких є саме DES, що робить її досить універсальною для написання коду алгоритмів шифрування. Інтеграція потребує ніяких додаткових налаштувань чи складних кроків, що дозволяє зосередитись безпосередньо на основному коді програми. Це дуже сильно економить час розробника та мінімізує вірогідність помилки.

Бібліотека є досить стабільною і розвиваючоюся, а тому її використання є цілком безпечним та надійним. А ще однією перевагою використання Crypto.js є її оптимізація швидкості та ефективності. Саме через цю оптимізацію можна дуже швидко шифрувати та дешифрувати текст навіть при значних обсягах даних. Якщо розглядати це з точки зору реалізації шифру DES, то швидкість обробки даних має велике значення через те, що цей алгоритм застосовують у додатках для збереження даних у зашифрованому вигляді, а тому система досить сильно навантажується.

Також алгоритми Crypto.js є безпечними та стабільними через те, що бібліотека відкрита, як і мова програмування, а тому розробники постійно підтримують її та додають нові стандарти безпеки. Тому можна бути впевненим в тому, що дані, котрі будуть оброблятися цією бібліотекою, будуть надійно захищені і за найсучаснішими технологіями.

Якщо врахувати всі вищеперелічені фактори, то можна зробити висновок, що бібліотека Crypto.js є гарним вибором для розробки методу шифрування DES та не лише його.

Метод DES є симетричним методом шифрування, а це означає, що він використовує ключ як для шифрування, так і для дешифрації. Особливість цього методу є тим, що для коректної роботи необхідно розбивати вхідний текст на блоки фіксованого розміру. За допомогою ключа шифрування та досить складних математичних операцій, шифрування цим методом є досить складним та багатокроковим процесом, що перетворює відкритий текст на зашифрований, захищений від стороннього доступу.

Для того щоб зашифрувати текст методом, необхідно виконати такі кроки:

- Розбиття тексту на блоки. Початковий текст розподіляється на блоки розміром 64 біти. Якщо останній блок коротший за 8 символів, то до нього додаються спеціальні символи, щоб зробити його потрібного розміру.

- Вибір ключа. Ключ вводиться з клавіатури, і він має бути довжиною 64 біти, але ефективно використовується лише 56, оскільки вісім із них призначені для перевірки парності. Цей ключ є основою процесу, оскільки саме він вирішує, яким способом дані будуть зашифровані.

- Початкова перестановка. Кожен блок даних проходить початкову перестановку, яка змінює порядок його бітів згідно із заздалегідь заготовленою таблицею. Ця операція не впливає на безпеку шифрування, але вона необхідна для того, щоб почати процес кодування для наступних етапів.

- Розділення на дві частини. Блок даних ділиться на дві рівні частини: ліву та праву. Кожна з цих частин обробляється окремо протягом 16 повторюваних раундів.

- Раундові обчислення. У кожному з 16 раундів права частина блока модифікується за допомогою ключа, котрий було отримано з клавіатури. Права частина комбінується з лівою за допомогою математичної операції XOR [20]. Потім ліва і права частини міняються місцями. Для забезпечення надійності використовується комбінація перестановок і підстановок.

- Об'єднання та завершальна перестановка. Після того як 16 раундів були пройдені, і ліві з правими частини об'єдналися, блок проходить через завершальну перестановку, яка знову змінює порядок бітів. Отримані блоки об'єднуються в один і на виході формують новий, повністю зашифрований текст.

Детальну роботу алгоритму шифрування методом DES зображено на схемі нижче (Рисунок 2.5).

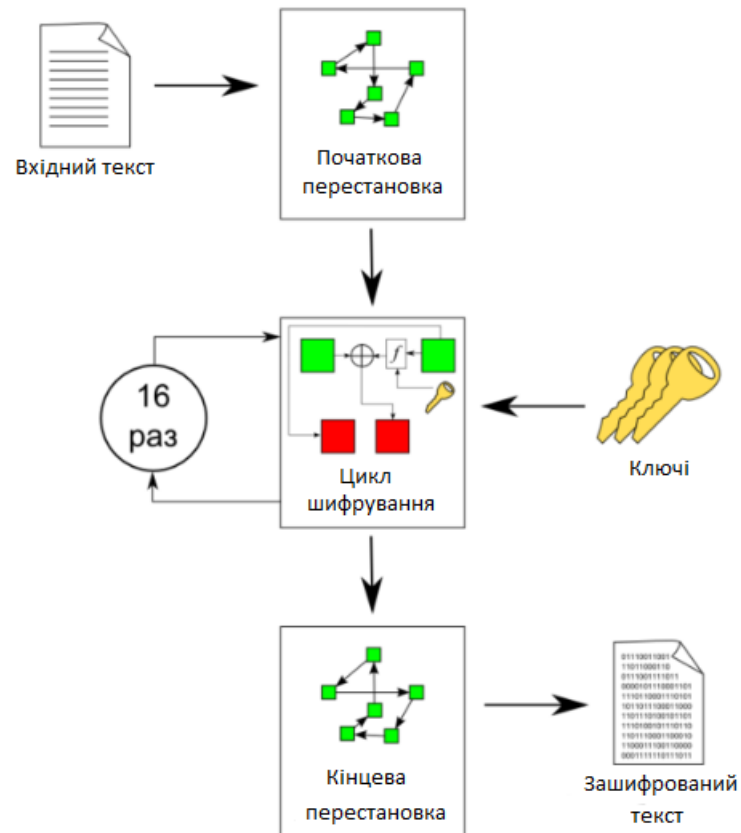


Рисунок 2.5 – схема алгоритму шифрування тексту методом DES

Розшифрування — це дзеркальний процес, який дозволяє отримати початковий текст із зашифрованого, використовуючи той самий ключ. Як було сказано раніше, метод DES є симетричним, а саме тому така опція є доступною.

Для виконання розшифрування тексту методом DES необхідно виконати такі кроки:

- Розбиття на блоки. Як і при шифруванні, зашифрований текст необхідно розподілити на блоки розміром 64 біти.
- Використання ключа. Ключ має бути тим самим, та він застосовується у зворотному порядку: ключ, використаний у першому раунді шифрування, стає останнім при розшифруванні, і так далі.

- Зворотні 16 раундів. Кожен блок проходить через 16 раундів, але всі операції виконуються навпаки. Ліва і права частини повинні помінятися місцями, використовуючи зворотні перестановки та підстановки. Математична операція XOR дозволяє повернути всі модифікації та перетворення, котрі були зроблені під час процедури шифрування.
- Зворотні перестановки. Після того як 16 раундів були пройдені, проводиться обернена фінальна перестановка, яка повертає біти до початкового порядку.
- Об'єднання блоків. Усі розшифровані блоки об'єднуються, щоб отримати початковий читабельний текст.
- Видалення заповнювачів. Якщо в процесі шифрування додавалися заповнювачі для коректного формату даних, то вони видаляються, щоб відновити текст до оригінального читабельного вигляду.

2.7 Реалізація методу AES

Метод AES (Advanced Encryption Standard) є модернізованою версією шифру DES та також реалізується за допомогою бібліотеки CryptoJS. Для того, щоб почати шифрування, необхідно виконати певні кроки:

- Поділ даних на блоки. Як і в методі DES, дані ділять на блоки. Але на відміну від попереднього методу, кожен блок має чіткий розмір 128 біт, тобто 16 символів. Якщо текст коротший за цю довжину, він доповнюється спеціальними заповнюючими, щоб набрати необхідної довжини. Окрім тексту, необхідний ще й ключ, що буде вводиться з клавіатури. Ключ може бути довжиною 128, 192 або 256 біт [21]. Складність процесу напряду залежить від довжини ключа. Чим довший ключ, тим більше раундів проходить шифрування: 10, 12, 14 раундів відповідно.

- Початкова операція (Add Round Key). На цьому етапі кожен символ тексту починає свою взаємодію із ключем шифрування. Тут застосовується, як і в оборонному методі, математична операція XOR (побітова "виключне або"), яка перемішує введені дані з ключем шифрування.

- Раундові операції. Кожен блок даних проходить серію перетворень, які називають раундами. Кожен раунд включає в себе чотири етапи, що виконуються послідовно. Перший етап SubBytes замінює кожен символ тексту на інший за допомогою спеціальної таблиці підстановок, відомої як S-Box. Ця таблиця виконує функцію словника шифрування, де кожен символ має свій шифрований відповідник. Другий етап ShiftRows [22]. Під час цього етапу символи, що були в блоках, змінюються на певну кількість позицій. Наприклад, символи в першому рядку залишаються на місці, в другому — зсуваються на одну позицію вліво, у третьому — на дві і так далі. Третій етап MixColumns (перемішування колонок). На цьому етапі всі дані в колонках змішуються за допомогою деяких математичних операцій. Останній етап Add Round Key (додавання ключа раунду). Знову застосовується математична операція XOR, але цього разу з іншим ключем, який створюється для кожного раунду за спеціальним алгоритмом. Кожен ключ походить на попередній, але виглядає зовсім інакше. Ці чотири етапи повторюються в кожному раунді. Саме тому кожен раунд надає додаткову безпеку до загального рівня захисту. Останній раунд містить усі вищеперераховані етапи, крім MixColumns. Після всіх цих перетворень вихідний текст перетворюється на зашифрований і стає абсолютно нечитабельним. Без ключа набір символів відновити неможливо.

Процес шифрування AES зображено на схемі нижче (Рисунок 2.6).

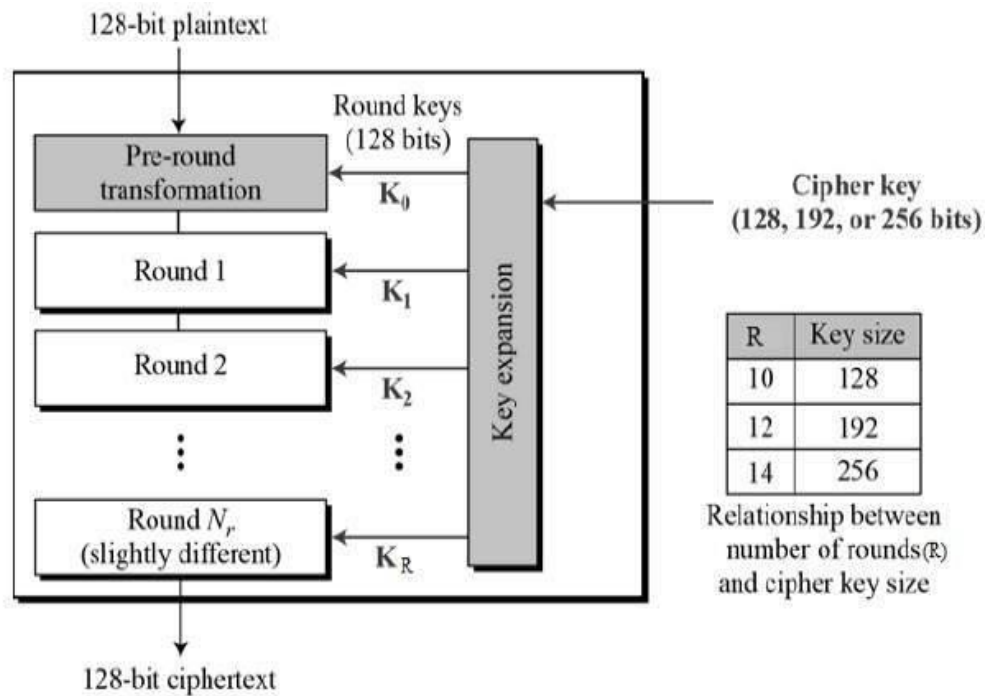


Рисунок 2.6 – Схема шифрування AES

Процес розшифровки — це повне зворотнє шифрування. Текст повертається до початкового вигляду, проходячи всі ці етапи, але в зворотному порядку. А для того, щоб розшифрувати текст, необхідно виконати такі кроки:

- Ініціалізація. Починається з дешифрування ключа останнього раунду. Без цього етапу неможливо відкрити "перший" рівень сейфа.
- Зворотні раундові операції. У кожному раунді виконуються зворотні операції:
 - Inverse ShiftRows: символи зсуваються назад до їх початкових місць;
 - Inverse SubBytes: символи повертаються до своїх початкових значень за допомогою зворотної таблиці підстановок (Inverse S-Box);
 - Inverse MixColumns: колонки знову змішуються, але цього разу так, щоб повернути початкову структуру даних;
 - AddRoundKey: дані знову XOR-ються з ключем відповідного раунду, щоб повернути їхній початковий вигляд.

- Зворотний початковий раунд. Наприкінці текст проходить через зворотну версію початкового раунду `AddRoundKey`. Це фінальний етап розшифрування.

2.8 Реалізація методу RC5

Наступним методом в реалізації є метод шифрування RC5. На відміну від AES і DES, RC5 дозволяє налаштовувати довжину ключа від 0 до 2040 біт та розмір блоку від 32 до 128 біт, що дає можливість адаптувати шифрування до конкретних потреб. Також алгоритм RC5 має досить просту реалізацію, його структура менш складна, ніж у AES. RC5 є симетричним шифром, котрий використовує блоки даних і ключі для того, щоб шифрувати текст [23]. Шифрування цим методом працює в кілька етапів. Початковий процес включає розподіл ключа на підготовче значення, створення масивів та виконання серії операцій шифрування.

Для того щоб шифрувати методом RC5, необхідно виконати такі пункти:

- Ключова процедура: Перше, що потрібно зробити для того, щоб розпочати шифрування, — це підготувати ключ. Ключ у RC5 передається як набір байтів, які розширюються в масив, що називається *S*. Розширення ключа включає кілька важливих операцій, таких як значення на основі попередніх результатів з'єднаних символів, включення цілих чисел та їх змішування через додавання та побітові операції XOR [24]. Кожна пара цих ключових значень обчислюється, і нові отримані значення записуються в масив *S*, котре містить необхідні параметри для проведення кожного раунду шифрування. Завдяки цим операціям кожен раунд шифрування сильно змінює вхідний текст, орієнтуючись на початковий ключ.

- Процес шифрування: Шифрування тексту в RC5 відбувається за декілька раундів, в кожному з яких застосовуються дві основні операції: XOR, що використовується для змішування даних із значеннями з масиву S, а додавання переміщує значення на базі попередніх операцій і додає отримане значення до тексту. Кожен раунд робить ці дві операції з парою блоків, і потім це результат використовується як вхід для початку наступного раунду. Кількість раундів залежить від обраної конфігурації та може варіюватися. У програмі реалізовано за замовчуванням 12 раундів.

- Формат вхідного тексту: Текст, введений з клавіатури, розбивається на блоки, кожен з яких складається з двох символів. Якщо текст не ділиться рівно на блоки, як і в попередніх методах, до нього додаються додаткові символи.

Нижче зображена схема роботи алгоритму RC5 (Рисунок 2.7).

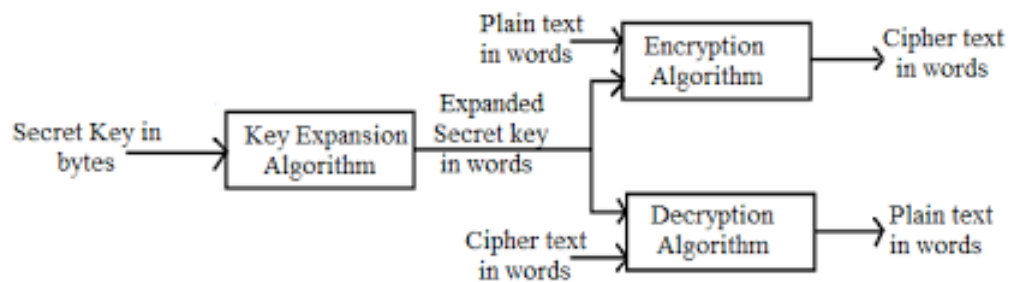


Рисунок 2.7 – Схема роботи алгоритму RC5

Щоб розшифрувати текст методом RC5, потрібно виконати аналогічні дії, але у зворотному порядку. Оскільки цей шифр є симетричним, основною різницею є те, що операція XOR і додавання виконується в зворотному напрямку — від останнього раунду до першого. Кожен зворотний раунд виконується в зворотному порядку: спочатку застосовуються операції, які були виконані в першому раунді шифрування, а потім поступово відбуваються попередні раунди, відновлюючи початкові дані. Додавання ключа з кожним раундом змінює ключ,

але це відбувається не як під час процесу шифрування, бо XOR застосовується в зворотному порядку. Важливими аспектами алгоритму RC5 є його гнучкість і налаштування кількості раундів, а також довжини ключа, що дозволяє забезпечити рівні безпеки для зовсім різних потреб. У коді програми шифрування та розшифрування реалізується за допомогою функції `decrypt`. Ключ підготовлюється через функцію `keySchedule`, яка створює масив `S` для кожного раунду.

2.9 Реалізація шифру Вернама

Шифрування і розшифрування методом Вернама базуються на використанні операції XOR між кожним символом відкритого тексту і відповідним символом ключа. Цей метод є одним з варіантів поточного шифрування, який є симетричним, що означає, що для шифрування і розшифрування використовується той самий ключ.

Шифрування методом Вернама працює наступним чином:

- Підготовка ключа: Для шифрування методом Вернама потрібно згенерувати випадковий ключ, довжина якого повинна бути такою ж, як і довжина тексту, що підлягає шифруванню. Ключ може бути будь-яким набором символів, зазвичай це набір випадкових символів [25].
- Шифрування: Кожен символ відкритого тексту (`plainText`) перетворюється на його ASCII код. Потім цей код піддається операції XOR з відповідним символом ключа (`key`). Операція XOR змінює значення кожного символу в залежності від його значення і відповідного символу в ключі. Наприклад, якщо символ відкритого тексту і ключа однакові, результат буде нулем (що створює непередбачуваний результат при шифруванні). Якщо ж

символи різні, результат буде іншим, що забезпечує складність дешифрування без правильного ключа.

- Результат шифрування: Кожен символ в результаті операції XOR перетворюється назад у символ за допомогою функції `String.fromCharCode()`, що дає зашифрований текст, який виглядає як рядок незрозумілих символів.

Процес розшифрування методом Вернама працює за такими кроками:

- Ідентичність операцій: Оскільки метод Вернама є симетричним, процес дешифрування є таким самим, як і процес шифрування. Текст, який був зашифрований методом XOR з певним ключем, можна розшифрувати тим самим ключем за допомогою такої ж операції XOR. Це відбувається через те, що операція XOR є своєрідною оберненою операцією: якщо $A \oplus B = C$, то $C \oplus B = A$. Таким чином, для того щоб отримати вихідний текст, достатньо ще раз застосувати XOR з тим самим ключем.

- Дешифрування: Тому для дешифрування просто застосовують функцію `vernamEncrypt()` з зашифрованим текстом і тим самим ключем. В результаті отримуємо оригінальний відкритий текст.

На схемі нижче зображено процеси шифрування та розшифрування тексту методом Вернама (Рисунок 2.8).

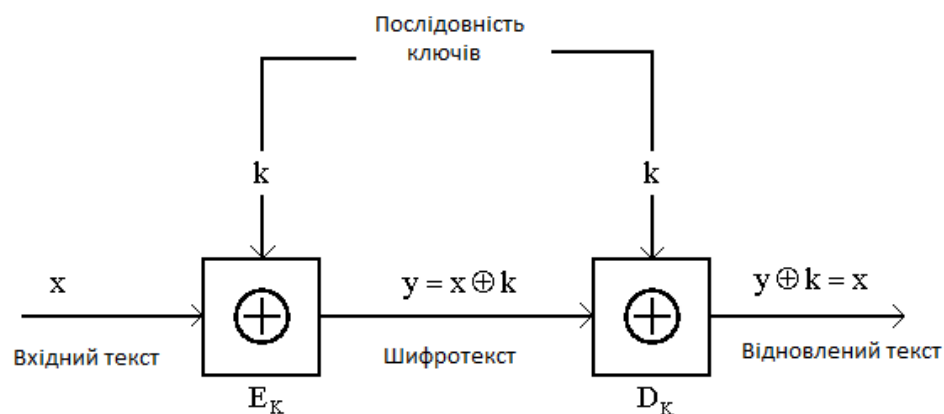


Рисунок 2.8 – Схема шифрування і розшифрування методом Вернама

Ключовим аспектом для надійного використання шифру Вернама є необхідність мати випадковий ключ, який має бути не коротшим за довжину зашифрованого тексту або рівним йому. Важливо також, щоб цей ключ використовувався лише один раз. Така практика гарантує непередбачуваність процесів шифрування і розшифрування, що робить метод дуже надійним для захисту переданих даних.

Висновки до розділу 2

Підводячи підсумки цього розділу, можна заявити, що було реалізовано всі перераховані методи шифрування, а саме: метод Цезаря, блоковий метод, метод DES, метод AES, метод RC5 та метод Вернама. Усі методи шифрування були реалізовані на мові програмування JavaScript, інтерфейс був реалізований за допомогою мови HTML та стилів CSS. Деякі методи були реалізовані за допомогою підключення бібліотеки CryptoJS до коду, а саме методи DES та AES. Усі методи відрізняються за складністю побудови та надійністю шифрування. Алгоритми роботи шифрування всіх реалізованих методів були представлені в текстовому та схематичному вигляді.

3 РЕАЛІЗАЦІЯ МЕТОДІВ ЗЛАМУ

Для того щоб протестувати надійність методів шифрування, необхідно розробити методи зламу, котрі будуть симулювати реальну загрозу крадіжки даних. Відповідно до кожного методу шифрування будуть розроблені методи, до яких вони найбільш уразливі. В цьому розділі буде продемонстровано принципи роботи методів зламу та способи їх реалізації.

3.1 Метод перебору

Метод перебору є одним із найпростіших методів зламу. Суть його полягає в тому, що він перебирає усі можливі варіанти і порівнює їх із вхідним текстом. В програмі реалізовано через перебір всіх можливих зсувів, щоб знайти правильний. Він застосовується для зламу метода Цезаря [26]. Оскільки шифр Цезаря передбачає використання фіксованого зсуву для зміщення букв, кількість можливих варіантів зсувів обмежена певним числом, що робить метод перебору ефективним для цього типу шифру.

Процес зламу відбувається наступним чином:

- Користувач вводить зашифрований текст.
- У функції `decryptButton` запускається метод зламу, який спробує всі можливі зсуви, починаючи з 0.
- В кожній спробі відбувається дешифрування за допомогою функції `caesarDecrypt`, яка використовує поточний зсув (спробу).
- Результат кожної спроби виводиться на екран через `breakOutput`, щоб користувач бачив, що відбувається на кожному етапі.

Алгоритм роботи зламу виглядає так:

- Для кожної спроби зламу застосовується зсув для дешифрування зашифрованого тексту.
- Якщо дешифроване повідомлення збігається з оригінальним текстом, залам вважається успішним, і процес припиняється.
- Якщо після 26 спроб правильний текст не знайдений, виводиться повідомлення про невдачу.

Метод перебору працює наступним чином:

- `decryptButton` ініціює процес зламу при натисканні кнопки.
- `caesarDecrypt` викликається з поточним зсувом (від 0 до 25), щоб спробувати всі можливі варіанти дешифрування.
- Якщо розшифрований текст збігається з оригінальним, процес зламу завершується і виводиться повідомлення про успіх.
- Кожна спроба виводиться на екран для користувача, щоб він міг спостерігати за прогресом.

Таким чином, метод перебору є ефективним інструментом для дешифрування шифру Цезаря завдяки обмеженій кількості можливих варіантів зсуву. Реалізація цього методу в програмному середовищі дозволяє користувачам легко спостерігати за процесом дешифрування та швидко знаходити правильний варіант розшифрованого тексту. Простота та прозорість алгоритму роблять його зручним для використання та розуміння навіть для початківців.

Алгоритм роботи методу перебору зображено на схемі (Рисунок 3.1)

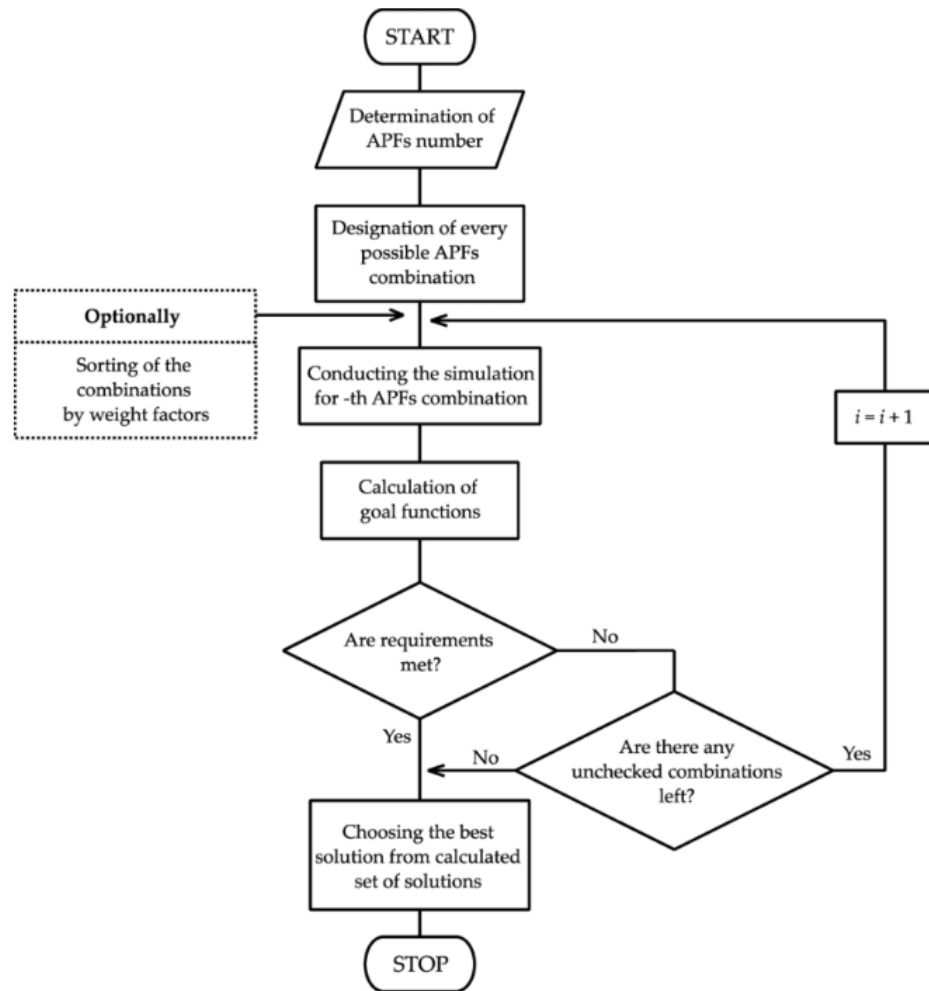


Рисунок 3.1 – Схема алгоритму роботи методу перебору

Схема наглядно ілюструє процес зламу за допомогою методу перебору. Метод є примітивним, а тому від підходить лише для зламу методів шифрування із слабкою криптографічною стійкістю.

3.2 Метод частотного аналізу

Метод частотного аналізу є одним із найстаріших методів зламу і найбільш ефективним підходом до зламу шифрів заміни. Він працює на основі

відслідковування частоти вживання букв або груп літер у тексті. Цей метод дозволяє розшифровувати зашифровані повідомлення, які використовують прості заміни, такі як шифр Цезаря або, як у цьому випадку, блоковий шифр.

Для того щоб зламати блоковий шифр методом частотного аналізу, необхідно виконати наступні кроки:

- Підготовка зашифрованого тексту. Необхідно отримати зашифрований текст, який буде підданий аналізу. Текст може бути представлений у вигляді послідовності літер або символів. Важливо, щоб він був достатньо довгим, оскільки на коротких фрагментах частотний аналіз є малоефективним.

- Розрахунок частоти символів. Для виконання частотного аналізу необхідно розрахувати частоту вживання кожного символу в тексті [27]. Це можна реалізувати шляхом підрахунку, скільки разів кожен символ з'являється в тексті. Потім, після підрахунку всіх частот символів, необхідно упорядкувати їх у порядку зменшення числових значень. Частотний аналіз має ігнорувати символи, що не є літерами, а саме пробіли, знаки пунктуації тощо.

- Побудова таблиці частот символів мови. Необхідно побудувати таблицю частот символів тієї мови, якою, ймовірно, написаний оригінальний текст. Частоти літер у природних мовах мають певні закономірності. Наприклад, в англійській мові найчастіше зустрічаються літери E, T, A, O, I, а в українській мові найчастіше зустрічаються літери O, I, E, T, H. Такі таблиці можна отримати на основі великого обсягу текстів, написаних обраною мовою.

- Співвіднесення символів. Після того як були обраховані частоти вживання літер у тексті, здійснюється співвіднесення символів. Наприклад, якщо у зашифрованому тексті зустрічається символ X, а в природній мові найбільш поширена буква E, можна припустити, що X відповідає E в оригінальному тексті. Така логіка діє і для інших символів — вони підбираються за цим частотним принципом.

- Заміна символів. Після того як символи будуть співвіднесені з відповідними символами мови, здійснюється заміна кожного символу на відповідний символ у вихідному тексті. На виході з'являється частково дешифрований текст. Якщо частотний аналіз виконано коректно, то велика частина тексту може бути зрозуміла навіть без повного дешифрування.

- Оцінка результату. Після дешифрування необхідно оцінити якість результату. Якщо текст має певну логіку та його можна прочитати і зрозуміти його сенс, то злам можна вважати успішним. У випадку, коли отриманий текст є нечитабельним та незрозумілим, можна спробувати інші варіанти співвіднесення частот.

- Корекція та повторне застосування методу. У деяких випадках може знадобитися повторне застосування цього методу з корекцією частот або підбором нових варіантів для рідкісних літер.

Частотний аналіз є одним із ключових методів криптоаналізу, особливо ефективним для шифрів, які не забезпечують достатнього рівня заплутаності. Окрім символів, часто аналізуються й біграми або триграми для глибшого розуміння структури тексту. Для підвищення ефективності аналізу можуть використовуватися статистичні моделі мови. Важливим аспектом є попередня нормалізація тексту, що включає видалення зайвих пробілів або спеціальних символів.

Процес роботи методу частотного аналізу продемонстровано на схемі (Рисунок 3.2). Це дозволяє встановити закономірності в шифрованому тексті, які можуть розкрити його структуру.

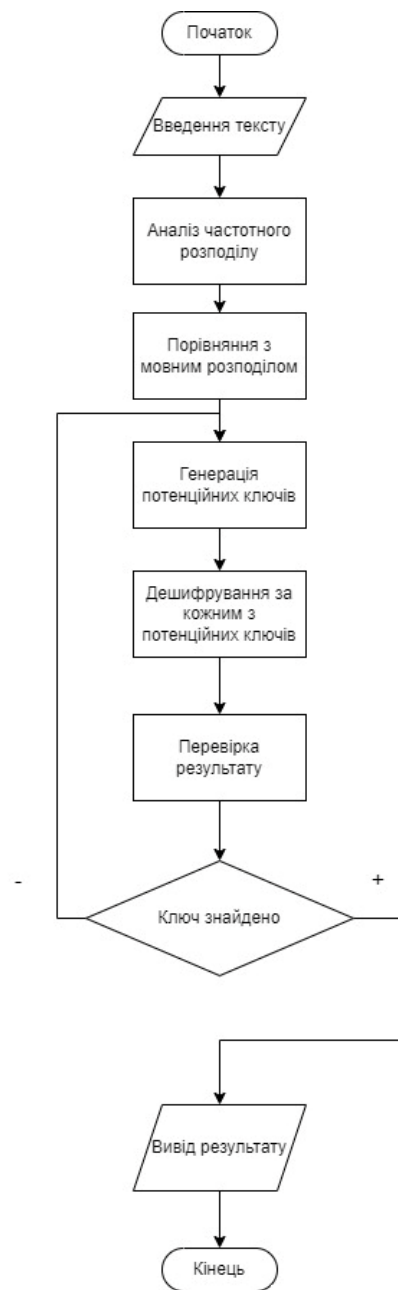


Рисунок 3.2 – Блок-схема методу частотного аналізу

Коли йдеться про складніші шифри, метод зламу частотного аналізу стає менш ефективним. У таких випадках він потребує використання додаткових методів для досягнення результату.

3.3 Метод пошуку ключа перебором

Метод пошуку ключа перебором ґрунтується на звичайній brute-force атаці. Але якщо для методу Цезаря здійснювався пошук зашифрованого тексту, то для зламу DES використовується пошук ключа [28]. У DES ключ обмежено певною довжиною, що робить перебір можливим, хоча й ресурсозатратним.

Програма перебирає всі можливі комбінації символів, які можуть утворити ключ. Кожна комбінація використовується для розшифрування. Якщо результат збігається з оригінальним текстом, пошук завершується.

Для виконання зламу необхідно виконати певні кроки:

- Генерація ключів. Функція `generateKeys` створює всі можливі комбінації ключів довжиною до 8 символів. Якщо довжина поточного ключа дорівнює максимально допустимій, виконується розшифрування. В іншому випадку функція додає до ключа кожен символ із набору `chars`, викликаючи себе рекурсивно для продовження побудови комбінацій.
- Розшифрування. Для кожного ключа викликається функція, яка приймає зашифрований текст і поточний ключ, а потім виконує спробу розшифрування тексту. Якщо розшифрований текст співпадає з оригінальним, ключ вважається знайденим, і процес зламу завершується.

Метод перебору ключа — це типовий приклад brute-force атаки. Він добре підходить для навчання або досліджень, але через велику складність не підходить для зламу сучасних систем шифрування. Однак для DES, де ключ має лише 56 біт, цей метод ще можна застосовувати, якщо є потужні обчислювальні ресурси для подальших обчислень.

3.4 Атака методом відкритого тексту

Атака методом відкритого тексту буде застосована до методу шифрування AES. Суть методу полягає в тому, що користувачу завчасно відомі кілька літер або слів відкритого тексту [29]. Програма аналізує співпадіння зашифрованого тексту з відомими літерами відкритого тексту, порівнює їх і підбирає правильний варіант.

Процес зламу описується наступними кроками:

- Користувач вводить відомий відкритий текст через інтерфейс (елемент з `id knownPlaintext`). Якщо текст не введено, виводиться попередження.
- Визначено максимальну кількість спроб (через лічильник `attempts`) і набір символів для перебору;
- Кнопка з `id hackBtn` ініціює процес зламу. Після натискання з'являється повідомлення про початок атаки.
- Створюється інтервал для регулярних спроб (кожні 10 мс).
- Функція `generateGuessedString` використовує кількість спроб (`attempts`) для генерації нових варіантів тексту:
 - Спочатку використовується відомий відкритий текст (`baseText`).
 - До цього тексту додаються символи з набору `chars` на основі поточної спроби.
 - Кожну нову спробу отримують шляхом обчислення залишку від ділення поточної кількості спроб на розмір набору символів.
 - Процес триває, поки кількість спроб не буде вичерпано або поки не буде досягнута максимальна довжина вгаданої строки.
 - Кожен згенерований текст порівнюється з оригінальним текстом (який не показано в коді, але ймовірно є заданим у змінній `originalText`).
 - Якщо вгаданий текст співпадає з оригіналом, процес зламу припиняється, і виводиться повідомлення про успіх.

- Якщо не співпало, виводиться поточна спроба з її результатом.

Чим більше частин відкритого тексту відомо користувачу, тим більше вірогідність успішного зламу шифру.

3.5 Метод диференційного криптоаналізу

Метод диференціального криптоаналізу є одним із найпотужніших методів атак на блочні шифри. Він дозволяє аналізувати, як зміни у відкритому тексті впливають на результат шифрування. Цей метод використовує різницю між двома відкритими текстами і аналізує, як ці зміни відображаються в зашифрованих текстах, щоб відновити ключ шифрування. В даній програмі цей метод використовується для перевірки методу RC5.

Для проведення такої атаки необхідно виконати кілька кроків:

- Вибір двох відкритих текстів. На першому етапі атаки потрібно вибрати два відкриті тексти, які будуть схожими. Наприклад, вони можуть відрізнятися лише в одному або кількох бітах. Це збереже основну структуру тексту і не зробить значних змін. Цю пару текстів надалі буде проаналізовано.
- Обчислення різниці між відкритими текстами. Після того як два тексти були обрані, потрібно визначити, які саме біти відрізняються між ними. Зміни в цих бітах будуть порівнюватися із змінами в результаті шифрування.
- Шифрування відкритих текстів. Обидва тексти шифруються за допомогою того самого ключа. Це дозволить отримати два зашифровані тексти, які потім будуть порівнюватися.
- Обчислення різниці між зашифрованими текстами. Після того як було виконано шифрування двох текстів за допомогою одного і того самого ключа, необхідно їх порівняти між собою. Це дозволить зрозуміти, як зміни вхідного тексту призвели до змін у результатах шифрування.

- Вивчення закономірностей між різницями. Після того як зміни були проаналізовані, криптоаналітик шукатиме закономірності, які дозволять йому зрозуміти, як саме вхідний текст впливає на зашифрований.
- Пошук вразливих місць у шифрі. Після того як все було проаналізовано, метод знаходить слабкі сторони шифру та намагається відновити ключ.
- Визначення частин ключа. Коли дані вже зібрані, криптоаналітик починає відновлювати ключ шифрування по частинам.
- Відновлення повного ключа. Продовжуючи аналіз, метод зламу буде відновлювати ключ до тих пір, поки не відновить його повністю. Це дозволить дешифрувати зашифрований текст без знання оригінального відкритого тексту.
- Дешифрування. Після того як всі частини ключа відновлені, можна використати його для подальшої дешифрації. Якщо текст співпадає з оригінальним, то дешифрування пройшло успішно [30].

Диференціальний криптоаналіз є потужним методом для зламу сучасних блочних шифрів, який базується на математичному аналізі змін у текстах. Реалізація цього методу в програмному середовищі дає змогу виявляти потенційні вразливості алгоритмів шифрування, таких як RC5, та проводити їх поглиблене дослідження, що сприяє підвищенню рівня криптографічної стійкості цих методів.

Роботу метода диференціального криптоаналізу можна побачити на схемі (Рисунок 3.3).

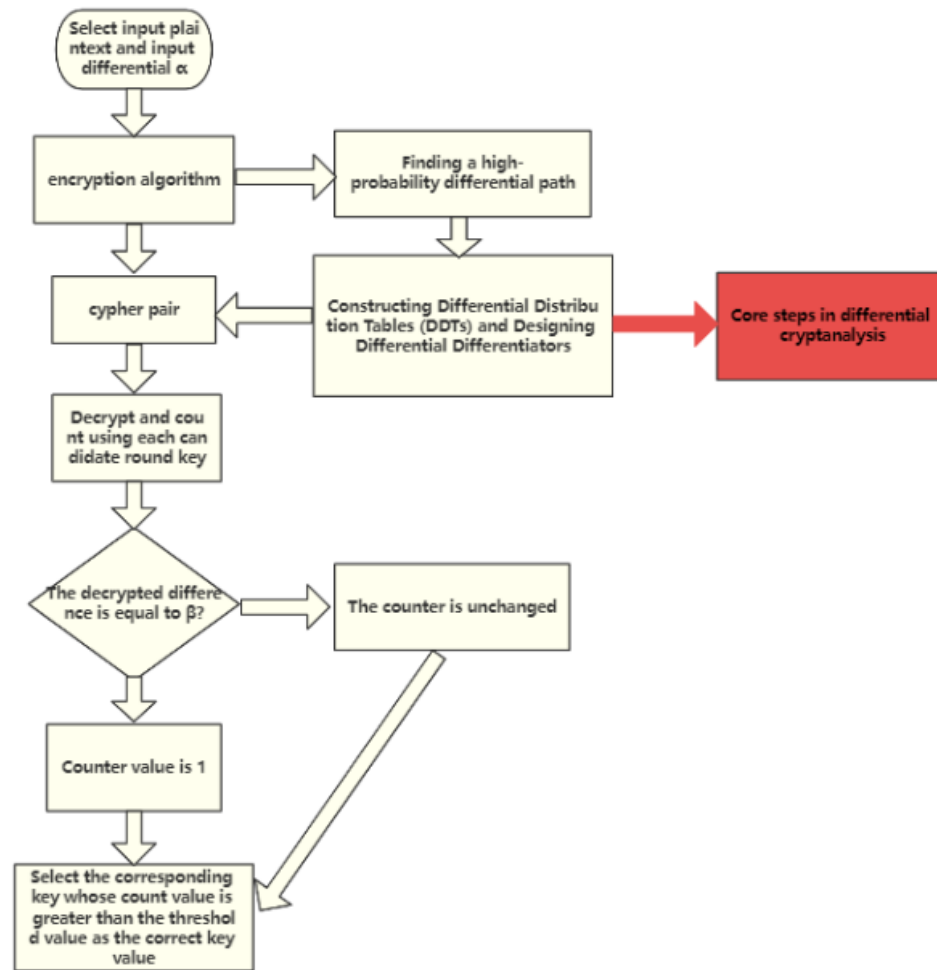


Рисунок 3.3 – Схема алгоритму методу диференціального криптоаналізу

Диференціальний криптоаналіз є одним із найсучасніших методів зламу, але цей метод вимагає використання того самого ключа двічі, а тому, якщо є задача уникнути потенційної загрози, однаковий ключ вживати не рекомендується.

Висновки до розділу 3

У даному розділі було продемонстровано, як було реалізовано методи зламу для подальшої оцінки криптографічної стійкості методів шифрування. Кожен метод був підібраний до конкретного методу шифрування, що дозволяє оцінити їх ефективність та обмеження. Метод перебору був розроблений для шифру Цезаря, оскільки цей шифр є найпростішим, що показує низький ступінь захисту цього методу шифрування. Метод частотного аналізу є потужним інструментом для зламу шифрів заміни, тому він був використаний для зламу блокового шифру. Метод пошуку ключа перебором показав свою здатність зламувати шифр DES, підбравши його ключ [31]. Атака методом відкритого тексту була використана проти методу шифрування AES, оскільки завчасно відома частина початкового тексту. Цей метод дозволяє зменшити кількість необхідних спроб і значно підвищує ймовірність успішного зламу. Метод диференціального криптоаналізу є одним з найпотужніших методів зламу для блочних шифрів, таких як RC5. Він дозволяє виявити закономірності між змінами у відкритому тексті та зашифрованому тексті, що допомагає відновити ключ. Усі методи зламу демонструють різний рівень ефективності в залежності від типу шифру, що свідчить про необхідність постійного вдосконалення систем шифрування для забезпечення високого рівня безпеки від сучасних загроз.

4 РОБОТА КОРИСТУВАЧА ІЗ СЕРЕДОВИЩЕМ ВИПРОБУВАННЯ КРИПТОГРАФІЧНОЇ СТІЙКОСТІ МЕТОДІВ ШИФРУВАННЯ

Для забезпечення комфортної та стабільної роботи середовища випробування криптографічної стійкості методів шифрування необхідно дотримуватися мінімальних системних вимог. Інтерфейс виконано в мінімалістичному стилі та спроектовано таким чином, щоб бути інтуїтивно зрозумілим для користувача. Тому проблем із взаємодією з програмою виникати не повинно. У цьому розділі буде описано, як взаємодіяти із середовищем, а також проведено обчислювальні експерименти.

4.1 Мінімальні вимоги системи для роботи із середовищем випробування криптографічної стійкості методів шифрування

Для комфортної взаємодії із середовищем необхідно мати пристрій, котрий буде відповідати мінімальним системним вимогам:

- Процесор Intel Core 2 duo;
- Об'єм оперативної пам'яті (RAM) – 512 МБ;
- Операційна система (одна з): iOS, Android, macOS, Windows, Linux.

На пристрої має бути встановлені такі програми:

- Інтернет браузер (один із): Google Chrome, Microsoft Edge, Opera, Safari, Mozilla Firefox.

4.2 Процес взаємодії із середовищем випробування криптографічної стійкості методів шифрування

Загальна структура взаємодії з програмою виглядає наступним чином (Рисунок 4.1) [51].

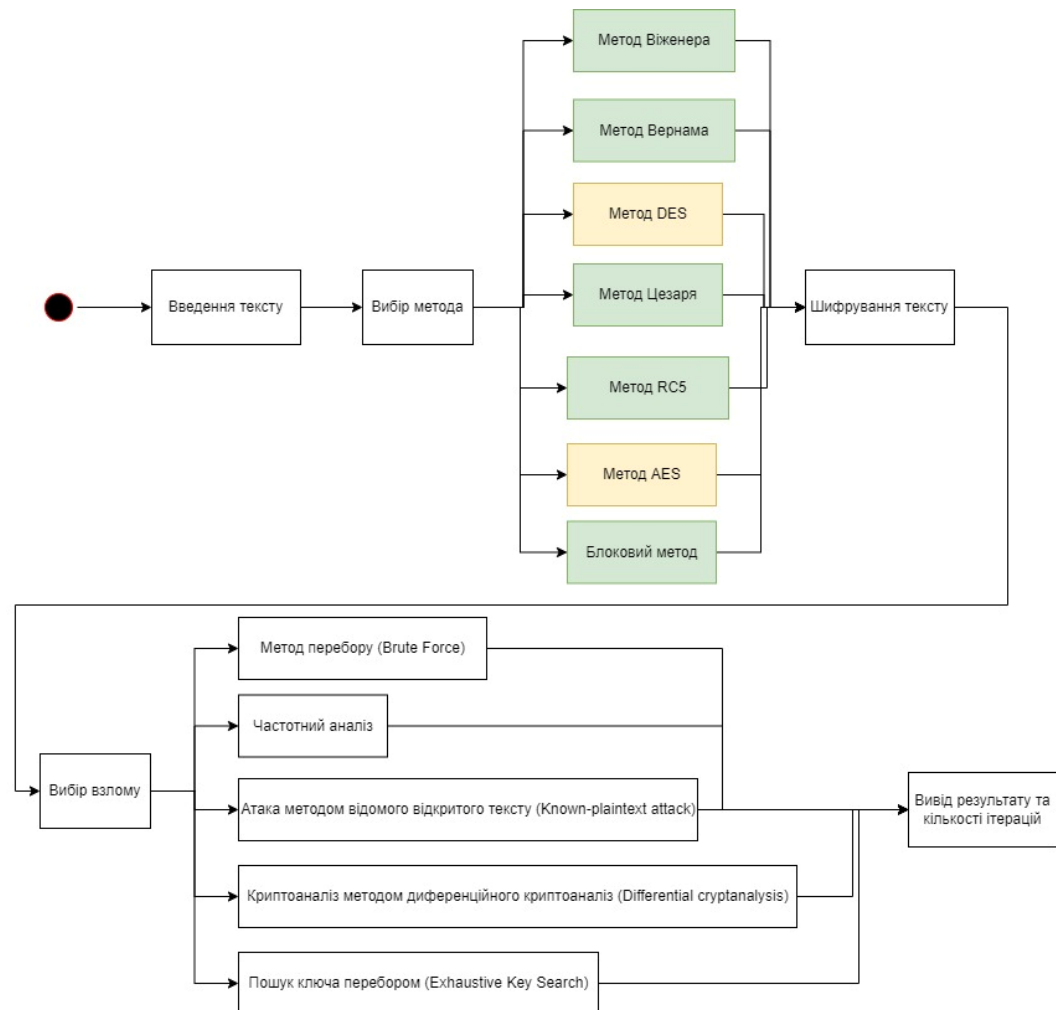


Рисунок 4.1 – Структура взаємодії із середовищем

Зеленим кольором відображені методи шифрування, що були розроблені власноруч, а помаранчевим – ті, що були реалізовані за допомогою бібліотек.

Якщо відкрити сторінку, то на екран виведеться повний інтерфейс середовища (Рисунок 4.2).

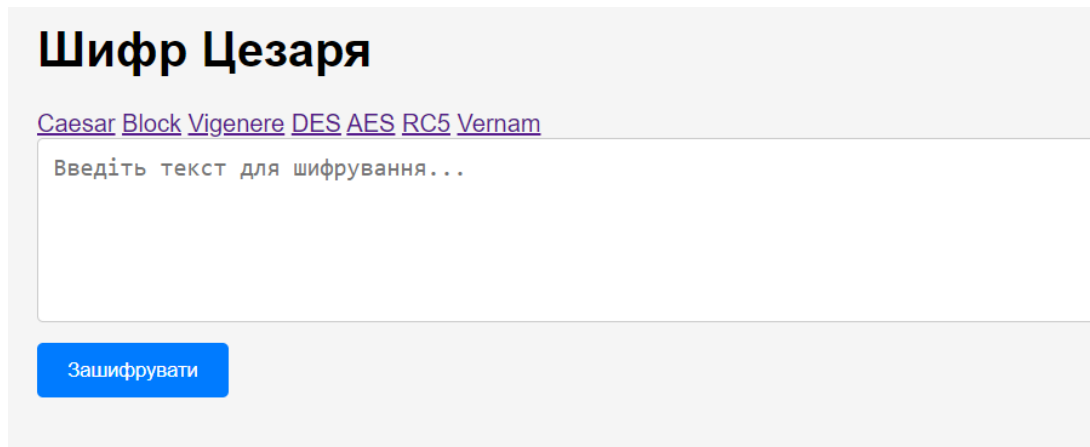


Рисунок 4.2 – Інтерфейс середовища

Зверху відображається обраний метод (Рисунок 4.3).

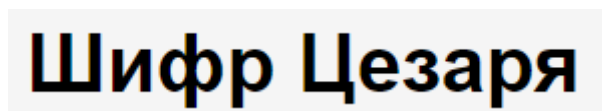


Рисунок 4.3 – Обраний метод

Під обраним методом знаходиться панель із вибором методу шифрування (Рисунок 4.4). При натисканні на назву методу відбудеться перехід на сторінку обраного методу.



Рисунок 4.4 – Панель вибору методу шифрування

За замовчуванням обирається метод Цезаря. Під вибором методу знаходиться поле для вводу тексту для шифрування (Рисунок 4.5).

Введіть текст для шифрування...

Рисунок 4.5 – Поле ввода текста

Після введення тексту (Рисунок 4.6) необхідно натиснути кнопку «Зашифрувати» (Рисунок 4.7) для шифрування тексту.

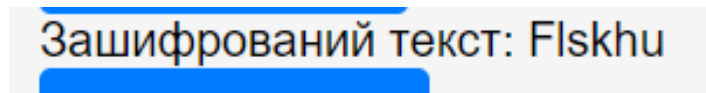
Cipher

Рисунок 4.7 – Заповнене поле для вводу тексту

Зашифрувати

Рисунок 4.8 – Кнопка «Зашифрувати»

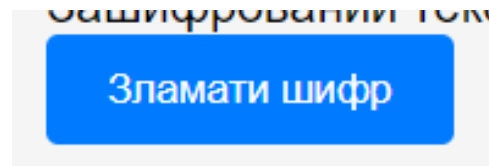
Після натискання на кнопку «Зашифрувати» відбувається процес шифрування і на екран виводиться шифротекст (Рисунок 4.9).



Зашифрований текст: Flskhu

Рисунок 4.9 – Зашифрований текст

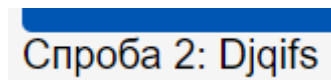
Коли текст зашифрований, то під ним з'являється кнопка «Зламати шифр» (Рисунок 4.10).



Зламати шифр

Рисунок 4.10 – Кнопка «Зламати шифр»

Після натискання на кнопку відбувається процес зламу (Рисунок 4.11).

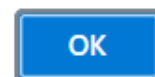


Спроба 2: Djqifs

Рисунок 4.11 – Процес зламу шифру

Як тільки шифр буде зламано, на екран виведеться відповідне повідомлення (Рисунок 4.12).

Текст розшифровано! Оригінал: CIPHER



ОК

Рисунок 4.12 – Повідомлення про успішний злам

При переході на сторінку блокового шифру, то додатково з'явиться кнопка «Дешифрувати» (Рисунок 4.13).

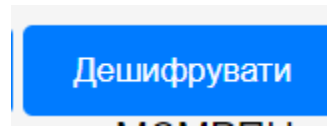


Рисунок 4.13 – Кнопка «Дешифрувати»

Якщо текст було зашифровано, то при її натискання текст буде розшифровано та виведено на екран (Рисунок 4.14).

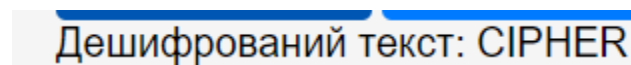


Рисунок 4.14 – Дешифрований текст

При натисканні кнопки «Зламати» почнеться процес злому шифру, під час якого на екран будуть виводитись всі ітерації процесу. Коли текст буде розшифровано, то відповідне повідомлення виведеться на екран (Рисунок 4.15).

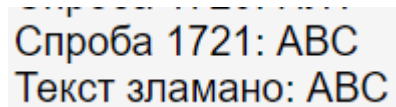
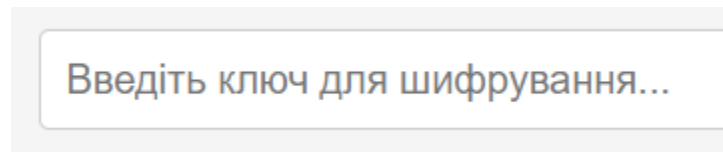


Рисунок 4.15 – Вивід ітерацій та результату зламу блокового шифру

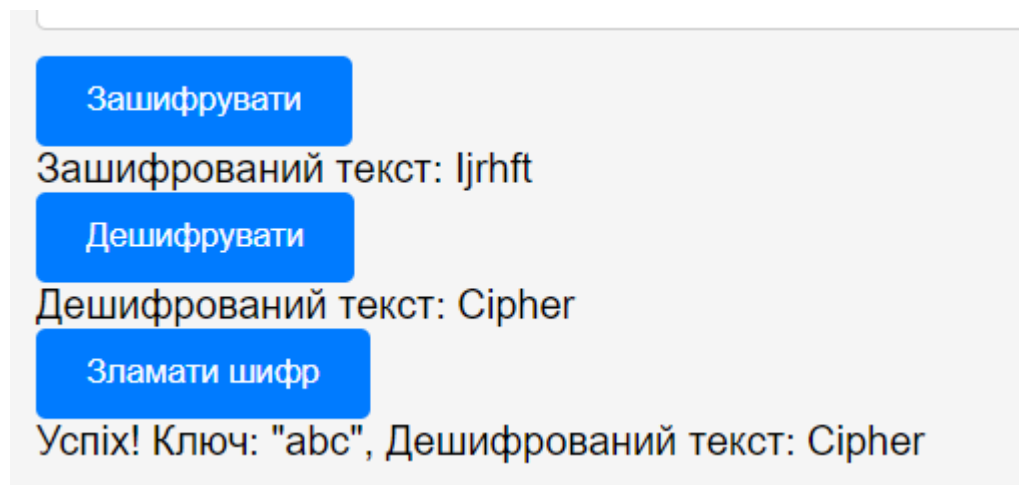
Якщо перейти на вкладку із шифром Віженера, то додатково з'явиться поле для вводу ключа шифрування (Рисунок 4.16).



Введіть ключ для шифрування...

Рисунок 4.16 – Поле для вводу ключа шифрування

Далі буде продемонстровано роботу всіх функцій методу Віженера (Рисунок 4.17).



Зашифрувати
Зашифрований текст: ljrhtf
Дешифрувати
Дешифрований текст: Cipher
Зламати шифр
Успіх! Ключ: "abc", Дешифрований текст: Cipher

Рисунок 4.17 – Демонстрація роботи шифру Віженера

Як можна побачити, при запуску зламу програма починає підбирати ключ. Якщо ключ підібрано правильно, то виводиться відповідне повідомлення.

Інтерфейс методу шифру DES схож на попередній, але всі ітерації підбору ключа виводяться на екран (Рисунок 4.18).

Cipher

Ключ (8 символів):

abcabcsab

Зашифрувати Дешифрувати Зламати

Зашифрований текст:
U2FsdGVkX19cA5yAlafWZe7RvHRe2Ql4

Дешифрований текст:
Cipher

Спробуємо ключ: аааааааа -
Спробуємо ключ: аааааааb -
Спробуємо ключ: ааааааас -

Рисунок 4.18 – Демонстрація роботи метода DES

При переході на вкладку із методом шифрування AES з'являється додаткове поле для вводу відомої частини тексту (Рисунок 4.19).

Введіть відомий відкритий текст...

Рисунок 4.19 – Поле для вводу відомого відкритого тексту

Нижче буде продемонстровано роботу метода AES (Рисунок 4.20).

Cipher

abcaabcabc

Ciph

Зашифрувати

Зашифрований текст: U2FsdGVkX18S1qOSICq6vjgTERDPa3zqqpVqRgDP71k=
 Дешифрований текст: Cipher
 Починається злам...
 Спроба 0: "Ciph"
 Спроба 1: "Ciphb"
 Спроба 2: "Ciphc"
 Спроба 3: "Ciphd"
 Спроба 4: "Ciphe"
 Спроба 5: "Ciphf"
 Спроба 6: "Ciphg"

Рисунок 4.20 – Демонстрація роботи метода AES

Якщо результат зламу буде успішним, то виведеться відповідне повідомлення (Рисунок 4.21).

Успіх! Знайдено збіг: Cipher

Рисунок 4.21 – Повідомлення про успішність зламу шифру AES

При переході на вкладку шифру RC5, ніяких додаткових полів не з'явиться. Інтерфейс схожий на попередні методи. Нижче буде продемонстровано роботу цього методу шифрування (Рисунок 4.22).

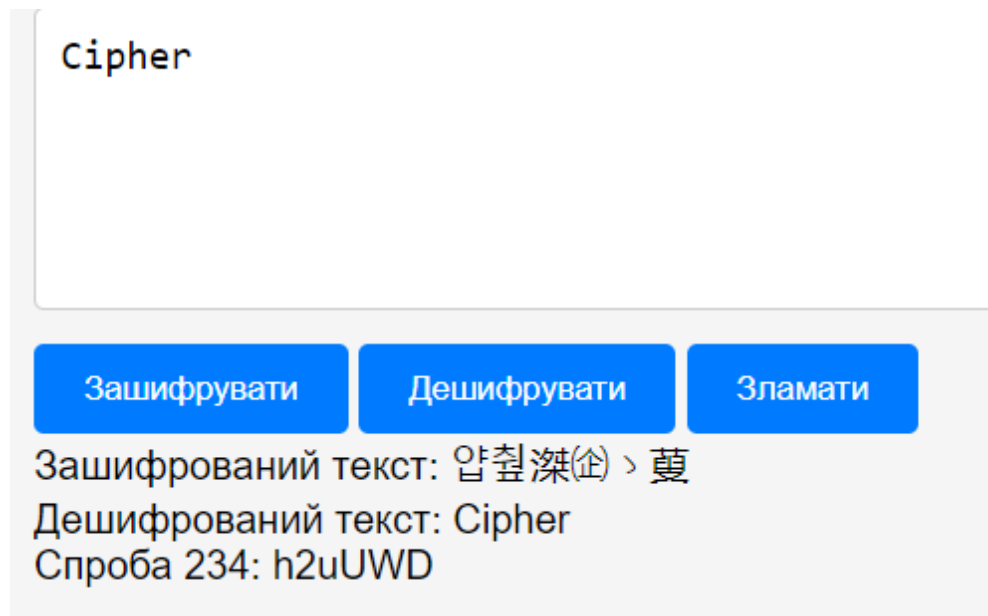


Рисунок 4.22 – Демонстрація роботи шифру RC5

При переході до шифру Вернама інтерфейс майже ідентичний методу Цезаря. Нижче показано, як працює цей метод (Рисунок 4.23).

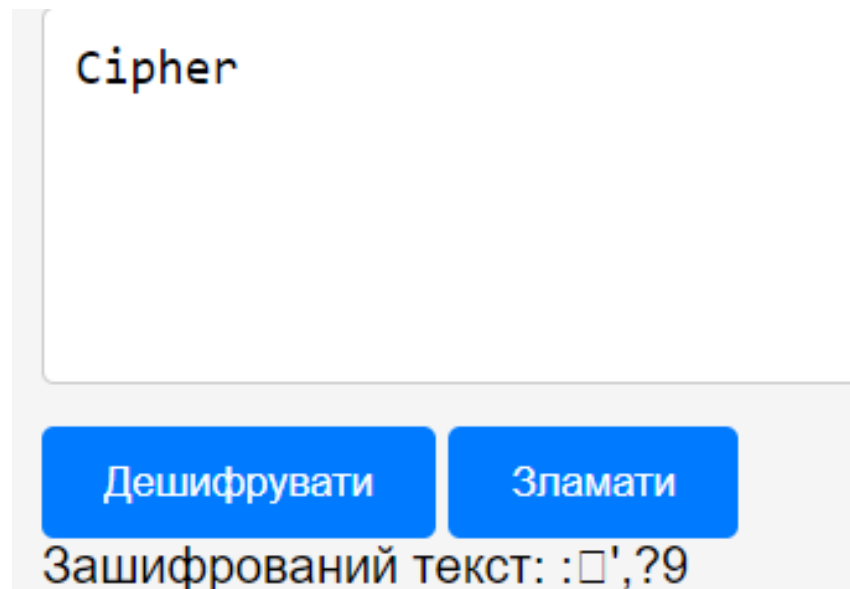


Рисунок 4.23 – Демонстрація шифрування методом Вернама

Таким чином, текст буде зашифровано так, що його неможливо буде зламати.

4.3 Криптоаналіз методів шифрування

Метод Цезаря є одним із найпростіших методів шифрування. Його головною вразливістю є обмежена кількість ключів, що дозволяє швидко підібрати правильний за допомогою методу перебору. Крім того, метод легко піддається частотному аналізу, оскільки розподіл символів у тексті після шифрування залишається незмінним. Це дозволяє зловмиснику зіставити частоти символів у зашифрованому тексті з типовими частотами символів у мові, що використовується.

Блоковий метод шифрування працює шляхом поділу тексту на блоки фіксованого розміру. Якщо блоки шифруються без додаткових операцій, таких як XOR, перестановки чи дифузія, то метод є вразливим до частотного аналізу. Використання однакових ключів для шифрування однакових блоків створює повторювані патерни, які можуть бути проаналізовані. Крім того, за недостатньої складності алгоритму можуть виникати слабкі місця, які полегшують злам методом диференційного криптоаналізу.

Шифр Віженера є розвитком шифру Цезаря, де використовується ключова фраза для створення різних зсувів для кожного символу. Основною його вразливістю є періодичність ключа. Якщо зловмисник знає довжину ключа, то шифр можна розбити на кілька підшифрів Цезаря, кожен із яких зламується окремо. Частотний аналіз є ефективним інструментом для зламу цього методу, особливо якщо текст довгий і повтори ключа очевидні.

Метод DES, попри свою популярність у минулому, має низку вразливостей, пов'язаних із короткою довжиною ключа (56 біт), що робить його вразливим до

методу перебору. Розширені методи аналізу, такі як диференційний і лінійний криптоаналіз, дозволяють ефективно знаходити вразливості у шифрі DES за умови наявності великої кількості зашифрованих даних. Крім того, однотипність ключів або неякісна генерація псевдовипадкових чисел може знижувати криптографічну стійкість.

Метод AES є одним із найстійкіших до криптоаналізу завдяки великій довжині ключа (128, 192 або 256 біт), сильній дифузії та складній нелінійній заміні. Однак метод відкритого тексту може застосовуватися до слабких реалізацій, де частина тексту відома зломиснику. Недоліки можуть виникати через неправильну імплементацію, наприклад, використання однакових ключів для різних блоків.

Метод RC5 є стійким до багатьох типів атак завдяки параметрам, які можна налаштовувати, таким як кількість раундів, розмір блоку та довжина ключа. Проте для малих кількостей раундів можливий успішний диференційний криптоаналіз. У складних конфігураціях RC5 залишається одним із найбільш надійних алгоритмів.

Шифр Вернама є теоретично абсолютним за умови використання абсолютно випадкового ключа, який не повторюється і має ту саму довжину, що і текст. Основна вразливість цього шифру пов'язана з практичними труднощами у створенні та передачі абсолютно випадкового ключа. Якщо ключ використовується повторно або є передбачуваним, шифр стає вразливим до атак, включаючи відомий текст і аналіз ключа.

Нижче представлена таблиця, що порівнює між собою всі ці методи за певними критеріями:

Таблиця 4.1 – Порівняльний аналіз існуючих криптографічних методів

Алгоритм и	Розмір ключа	Розмір блоку	Цикл и	Структура	Гнучк ий	Особливос ті
DES	64 біти	64 біти	16	Фейстель	Ні	Недостатньо сильний
AES	Від 8 до 256 біт	128 бітів	14	Фейстель	Ні	Дуже надійний та повільний
Цезаря	-	-	1	Перестановка	Ні	Абсолютно ненадійний та швидкий
Блоковий	-	64 біти	1	XOR	Ні	Абсолютно ненадійний та швидкий
Віженера	-	-	1	Перестановка	Ні	Абсолютно ненадійний та швидкий
RC5	Змінний	40- 2048	256	Ряд Фейстеля	Так	Швидкий шифр
Вернама	Від 1 до нескінченності	128 бітів	-	Фейстель	-	Абсолютна надійність та повільний

Згідно із таблицею, найслабшим за рівнем безпеки виявився метод Цезаря, а найбезпечнішим – шифр Вернама.

Висновки до розділу 4

У цьому розділі були розглянуті системні вимоги для комфортної роботи із середовищем випробування криптографічної стійкості методів шифрування, а також описано правильну взаємодію з програмою. Крім того, було проведено криптоаналіз усіх розглянутих методів шифрування. Найнадійнішим виявився метод Вернама, тоді як найбільш вразливим до атак був метод Цезаря. Метод RC5 та AES знаходяться на задовільному рівні безпеки. Метод DES є застарілим, але все ще використовується. Шифр Віженера та блоковий метод трохи перевершують шифр Цезаря за рівнем захисту, але все ще вважаються ненадійними.

5 РОЗРОБКА СТАРТАП ПРОЄКТУ

Середовище випробування криптографічної стійкості методів шифрування є унікальним, оскільки надає користувачеві не лише можливість шифрувати й дешифрувати текст, але й вибір методу для цього, а також перевірку його стійкості до зламу.

5.1 Опис ідеї стартап–проекту

В межах підпункту слід проаналізувати та подати у вигляді таблиць:

- зміст ідеї (що пропонується);
- можливі напрямки застосування;
- основні вигоди, що може отримати користувач товару;
- чим відрізняється від існуючих аналогів та замінників.

Перші три пункти подаються у вигляді таблиці (таблиця 5.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки.

Таблиця 5.1 – Опис ідеї стартап–проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Надання можливості користувачу шифрувати свої повідомлення та розшифровувати	1. Шифрування та розшифрування повідомлень за обраними методами	1. Надійний захист інформації від несанкціонованого доступу

Таблиця 5.1 (продовження)

отримані повідомлення за обраними методами шифрування та випробовувати їх на вразливість до зламу	2. Тестування методів шифрування на вразливість до зламу	2. Безкоштовний доступ на всіх платформах до захисту інформації
---	--	---

Аналіз потенційних техніко–економічних переваг ідеї (чим відрізняється від існуючих аналогів та замінників) порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко–економічних властивостей та характеристик ідеї;
- визначення попереднього кола конкурентів (проектів–конкурентів) або товарів–замінників чи товарів–аналогів, що вже існують на ринку, та проводиться збір інформації щодо значень техніко–економічних показників для ідеї власного проекту та проектів–конкурентів відповідно до визначеного вище переліку;
- проводиться порівняльний аналіз показників: для власної ідеї визначаються показники, що мають а) гірші значення (W, слабкі); б) аналогічні (N, нейтральні) значення; в) кращі значення (S, сильні) (таблиця 5.2) [43]

Таблиця 5.2 – Визначення сильних, слабких та нейтральних характеристик

No п/п		(потенційні) товари/концепції конкурентів	
		Мій проект	Cloaker
1	W слабка сторона	Призначена лише для шифрування за обраними методами	Дозволяє додавати методи власноруч

Таблиця 5.2 (продовження)

2	S сильна сторона	Простий в користуванні	Потребує спеціальних знань
3		Безкоштовний	Може собі дозволити лише велика компанія
4		Мультиплатформенний	Доступний лише на Windows

5.2 Технологічний аудит ідеї стартап–проекту

В межах даного підрозділу необхідно провести аудит технології, за допомогою якої можна реалізувати ідею проекту. Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (таблиця 5.3):

- за якою технологією буде виготовлено товар згідно ідеї проекту;
- чи існують такі технології, чи їх потрібно розробити/доробити;
- чи доступні такі технології авторам проекту.

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Нативний інтерфейс користувача для системи Windows	HTML, CSS	Наявна	Доступна безкоштовно

Таблиця 5.3 (продовження)

2	Шифрування тексту за обраними методами	JavaScript	Наявна	Доступна безкоштовно
3	Генерація ключів	CryptoJS	Наявна	Доступна безкоштовно
Висновок: проект реалізувати можливо. Обрані технології реалізації ідеї проекту: нативний інтерфейс користувача для будь-якої системи, шифрування тексту за обраними методами, генерація ключів.				

За результатами аналізу таблиці робиться висновок щодо можливості технологічної реалізації проекту: так чи ні, а також технологічного шляху, яким це доцільно зробити (з поміж названих технологій обираються такі, що доступні авторам проекту та є наявними на ринку).

5.3 Аналіз ринкових можливостей запуску стартап–проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів–конкурентів.

Спочатку проводиться аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таблиця 5.4).

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап–проекту

№ п/ п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	0
2	Загальний обсяг продаж, грн/ум.од	0 грн
3	Динаміка ринку (якісна оцінка)	–
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	–

Середня норма рентабельності в галузі (або по ринку) порівнюється із банківським відсотком на вкладення. За умови, що останній є вищим, можливо, має сенс вкласти кошти в інший проект.

За результатами аналізу таблиці робиться висновок щодо того, чи є ринок привабливим для входження за попереднім оцінюванням.

Ці дані допомагають окреслити загальні перспективи розвитку бізнесу на вибраному ринку. Додатково аналізуються ризики, пов'язані із виходом на ринок, та потенційні вигоди, які можуть бути отримані при успішній реалізації проекту. На основі цього формується стратегія подальших дій для максимізації прибутковості.

Надалі визначаються потенційні групи клієнтів, їх характеристики, та формується орієнтовний перелік вимог до товару для кожної групи (таблиця 5.5).

Таблиця 5.5 – Характеристика потенційних клієнтів стартап–проекту

№ п/ п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Необхідність у надійному захисті своїх даних від несанкціонова ного доступу	Армія, банки, енергетичні компанії	Особливості купівлі: представництва армії заключають довготривалі договори, а енергетичні компанії та банки віддають перевагу пробному терміну використання. Для армії важливо мати необмежений доступ до ресурсу, а для енергетичних компаній та банків – швидке застосування.	стабільність роботи, невисока ціна, наявність пробного періоду, наявність документац ії

Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (таблиці 5.6–5.7).

Таблиця 5.6 – Фактори загроз

No п/ п	Фактор	Зміст загрози	Можлива реакція компанії
1	Залежить від додаткових компонентів	Для коректної роботи середовища з надання послуг шифрувань потрібно встановити відповідний браузер, що підтримує JavaScript.	Перейти за відповідним посиланням для встановлення необхідного браузера.

Таблиця 5.7 – Фактори можливостей

No п/ п	Фактор	Зміст можливості	Можлива реакція компанії
1	Використання CryptoJS для генерації ключів шифрування.	Унікальний ключ шифрування можна згенерувати за допомогою бібліотеки CryptoJS.	Розширена підтримка існуючих платформ.

Надалі проводиться аналіз пропозиції: визначаються загальні риси конкуренції на ринку. Аналіз пропозиції необхідно виконати аналізуючи існуючі види конкуренцій.

Пропозиції повинні відповідати на питання “Як просувати продукт”. Аналіз пропозицій зображено на таблиці 5.8.

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції – монополія/олігополія/ монополістична/чиста	чиста	Прямі договори з стартапами, презентація продукту на виставках
2. За рівнем конкурентної боротьби – локальний/національний/...	національний	Публікація статей на міжнародних сайтах
3. За галузевою ознакою – міжгалузева/ внутрішньогалузева	внутрішньогалузева	Розвивати напрямки, нерозвинуті конкурентами
4. Конкуренція за видами товарів: – товарно–родова – товарно–видова – між бажаннями	товарно–видова	Розповідати про свої переваги перед конкурентом у цій галузі
5. За характером конкурентних переваг – цінова / нецінова	нецінова	Надання функцій, які не надають конкуренти

Таблиця 5.8 (продовження)

6. За інтенсивністю – марочна/не марочна	марочна	Надання функцій, які не надають конкуренти
---	---------	--

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі (таблиця 5.9) [44].

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари–замінники
	Відсутні	Програмний комплекс Cloaker	Мінімізація витрат часу постачальників	Контроль якості	Лояльність споживачів
Висновки:	Конкурента на боротьба відсутня	Є можливості виходу на ринок, оскільки існуючі рішення не надають потрібних переваг	Постачальники підлаштовуються під ринок	Клієнти диктують вимоги згідно з умовами експлуатації	Обмеження для роботи на ринку через товари замінники

На основі аналізу конкуренції, проведеного в п. 3.5 (таблиця 5.9), а також із урахуванням характеристик ідеї проекту (таблиця 5.2), вимог споживачів до

товару (таблиця 5.5) та факторів маркетингового середовища (таблиця 5.6–5.7) визначається та обґрунтовується перелік факторів конкурентоспроможності. Аналіз оформлюється за таблицею 5.10 [45].

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№ п/ п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Застосування сучасних засобів програмування	Існуючий програмний комплекс Cloaker реалізований мовою Fortran, яка на даний час не підтримується фірмою–розробником.
2	Простота роботи з програмою	Існуючий програмний комплекс Cloaker вимагає суттєвих специфічних знань
3	Безкоштовний доступ	Існуючий програмний комплекс Cloaker вимагає постійного оновлення підписки

За визначеними факторами конкурентоспроможності (таблиця 5.10) проводиться аналіз сильних та слабких сторін стартап–проекту (таблиця 5.11) [46]. На основі цього аналізу формується перелік заходів для підсилення слабких сторін та використання сильних сторін на користь проекту. Особливу увагу приділяють аспектам, які можуть забезпечити конкурентну перевагу на ринку. Також проводиться порівняння з конкурентами, щоб визначити можливості для покращення позицій проекту.

Таблиця 5.11 – Порівняльний аналіз сильних та слабких сторін

No п/ п	Фактор конкурентоспроможності	Бали 1– 20	Рейтинг товарів–конкурентів у порівнянні з середовищем випробування криптографічного захисту методів шифрування (даним продуктом)						
			–3	–2	–1	0	1	2	3
1	Інтерфейс користувача	15	+						
2	Контроль вхідних даних	10			+				
3	Застосування сучасних засобів програмування (.NET 4.6, HTML, CSS, JavaScript)	5	+						
4	Генерація унікальних ключів шифрування (CryptoJS)	15	+						

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT–аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (таблиця 5.12) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін (таблиця 5.11) [47].

Таблиця 5.12 – SWOT–аналіз стартап–проекту

Сильні сторони: Застосування сучасних засобів програмування (.NET 4.6, HTML, CSS, JavaScript). Генерація унікальних ключів шифрування (CryptoJS)	Слабкі сторони: Недостатній набір тестових сценаріїв для сертифікації підприємства.
Можливості: Відсутність повноцінних альтернатив.. Можливість подальшого супроводження та розвитку проекту.	Загрози: Неможливо пройти сертифікацію підприємства.

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів. [48].

Таблиця 5.13 – Альтернативи ринкового впровадження стартап–проекту

№ п/ п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Орієнтація середовища щодо застосування в армії	80 %	2 роки

Таблиця 5.13 (продовження)

2	Орієнтація сереловища щодо застосування в енергетичних компаніях	20 %	1 рік
3	Орієнтація сереловища щодо застосування в банках	50%	1 рік

Альтернатива, де отримання ресурсів є більш простим та ймовірним – №4 "Переорієнтація на генерацію серверної частини", що становить 80 відсотків. Це значення перевищує інші альтернативи.

Альтернатива, де строки реалізації є більш стислими – №1 " Орієнтація поточної моделі на ринок стартаперів". Терміни реалізації в цьому разі становлять лише 40 годин.

5.4 Розроблення ринкової стратегії стартап–проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис основних та цільових груп потенційних споживачів (таблиця 5.14).

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку [49].

Далі проводиться детальний аналіз потреб цільових груп, який дозволяє сформулювати ключові переваги товару для кожного сегмента. Це допомагає створити конкурентоспроможну пропозицію, адаптовану до вимог ринку.

На основі отриманих даних розробляється план маркетингових заходів, спрямованих на привернення уваги цільової аудиторії. Крім того, оцінюються ризики та можливості для успішної реалізації стратегії на обраному ринку.

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

No п/ п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Армія	Готові	Високий	Відсутня	Просто
2	Енергетичні компанії	Потребують переговорів	Середній	Відсутня	Просто
3	Банки	Потребують переговорів	Середній	Відсутня	Просто
Які цільові групи обрано: Армія					

Для роботи в обраних сегментах ринку необхідно сформулювати базову стратегію розвитку, яка враховує специфіку потреб цільових споживачів, особливості конкурентного середовища та ресурси, доступні для реалізації проекту (таблиця 5.15).

Таблиця 5.15 – Визначення базової стратегії розвитку

No п/ п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентос– проможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1	Орієнтація поточної моделі на ринок армії	Стратегія концентрованого маркетингу	Армія потребує надійні та унікальні ключі шифрування (CryptoJS) для захисту наданої та отриманої інформації.	Стратегія спеціалізації (спирається на диференціацію)

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів.

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту.

Наступним кроком є вибір стратегії конкурентної поведінки (таблиця 5.16).

Таблиця 5.16 – Визначення базової стратегії конкурентної поведінки

No п/ п	Чи є проект «першопрохідце м» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
1	Так	шукати нових споживачів	Ні	Стратегія заняття конкурентної ніші

З обраних сегментів до постачальника (стартап–компанії) та до продукту розробляється стратегія позиціонування (таблиця 5.17), що полягає у формуванні ринкової позиції, за яким споживачі мають ідентифікувати торгівельну марку чи проект [50].

Стратегія позиціонування враховує ключові переваги продукту, які відрізняють його від конкурентів, та визначає основні меседжі для комунікації зі споживачами.

Процес позиціонування включає дослідження ринкових трендів, аналіз конкурентів і визначення унікальної пропозиції цінності.

Успішне позиціонування дозволяє стартапу зайняти стійку ринкову нішу, забезпечуючи впізнаваність бренду та лояльність клієнтів.

Завдяки цьому формується чіткий образ компанії та продукту в свідомості цільової аудиторії, що сприяє зростанню продажів і довгостроковому розвитку.

Таблиця 5.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкуренто– спроможні позиції власного стартап– проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Стабільність роботи Невисока ціна Наявність пробного періоду Наявність документації	Стратегія спеціалізації (спирається на диференціаці ю)	Стабільність роботи та гнучкість в налаштуваннях системи забезпечується її модульністю. Генерація ключів шифрування (CryptoJS).	Стабільність роботи, наявність пробного періоду

5.5 Розроблення маркетингової програми стартап–проекту

Для цього у таблиці 5.18 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару [49]. Це дозволить чітко визначити унікальні переваги продукту, які забезпечать йому конкурентну перевагу на ринку. На основі узагальнених даних формулюються рекомендації щодо вдосконалення характеристик продукту та адаптації його до потреб цільової аудиторії.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

Но п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Універсальний доступ	Використання на всіх можливих платформах	Програмний комплекс Cloaker не підтримує мультиплатформну роботу
2	Простота використання	НЕ потребує спеціальних знань для роботи	Програмний комплекс Cloaker потребує специфічних знань для роботи.

Надалі розробляється трирівнева маркетингова модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (таблиця 5.19).

Таблиця 5.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за здумом	Середовище випробування криптографічного захисту методів шифрування		
II. Товар у реальному виконанні	Властивості/харак теристики	М/Нм	Вр/Тх /Тл/Е/Ор
	можливість оптимізації	М	Тл

Таблиця 5.19 (продовження)

	витрат часу		
	можливість оптимізації витрат коштів	М	Вр
	відповідність актуальним технологіям	Нм	Тх
	Відповідає міжнародним вимогам ICAO DOC 9911.		
	Пакування: готовий до використання файл		
	Марка: CryptoAndroid		
	III. Товар із підкріпленням	Потенційний користувач може ознайомитись з поточним товаром з наукових публікацій у наукометричних та фахових виданнях, а також наукових конференцій, на яких була представлена інформація про даний продукт	
За рахунок чого потенційний товар буде захищено від копіювання: Назва і контент захищені ліцензією MIT; захист інтелектуальної власності			

М/Нм – монотонні або немонотонні;

Вр/Тх/Тл/Е/Ор – вартісні, технічні, технологічні, ергономічні або органолептичні (останній – для продуктів харчування)

Після формування маркетингової моделі товару слід особливо відмітити – чим саме проект буде захищено від копіювання. Захист може бути організовано за рахунок захисту ідеї товару (захист інтелектуальної власності), або ноу–хау,

чи комплексне поєднання властивостей і характеристик, закладене на другому та третьому рівнях товару [58].

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (таблиця 5.20).

Таблиця 5.20 – Визначення меж встановлення ціни

№ п/ п	Рівень цін на товари– замінники	Рівень цін на товари– аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	0 грн	0 грн	98000...200000 грн	500000...1000000 грн

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (таблиця 5.21):

- проводити збут власними силами або залучати сторонніх посередників (власна або залучена система збуту);
- вибір та обґрунтування оптимальної глибини каналу збуту;
- вибір та обґрунтування виду посередників.

Крім того, необхідно враховувати специфіку ринку та особливості цільової аудиторії, що дозволить забезпечити ефективну доставку товару чи послуги. Вибір стратегії збуту також залежить від бюджетних можливостей компанії та її стратегічних цілей на різних етапах розвитку.

Таблиця 5.21 – Формування системи збуту

№ п/ п	Специфіка закупівельної поведінки цільових клієн- тів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Клієнт повинен надаватися в режимах “тріал” та “повний”. Сплата має проходити через банк.	Простота у встановленні програмного продукту, наявність документації по роботі з програмним забезпеченням	4: Розробник даного продукту – Веб-сайт – Користувач.	Проводити своїми силами.

Останньою складовою маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 5.22). Концепція передбачає вибір каналів комунікації, які найефективніше досягають цільову аудиторію, та визначення ключових повідомлень, що акцентують увагу на перевагах продукту. Успішні маркетингові комунікації сприяють формуванню довіри до бренду та стимулюють попит на продукцію. Важливим аспектом є інтеграція традиційних і цифрових інструментів для забезпечення максимальної охоплення аудиторії. Для цього використовується системний підхід, що дозволяє узгодити різні елементи комунікаційної стратегії. Це сприяє побудові

довготривалих відносин із клієнтами та створенню стійкої конкурентної переваги.

Таблиця 5.22 – Концепція маркетингових комунікацій

№ п/ п	Специфіка поведінки цільових клієнтів	Канали комунікації, якими користують ся цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Придбання програмного комплексу через офіційний сайт або зв'язавшись з представником.	Веб-сайти, менеджери	Модульність, гнучкість налаштувань, візуалізація даних	Довести, що програмний продукт здійснює надійний захист інформації	Шифрування та дешифрування повідомлень із тестуванням шляхом грубого взлому

Таким чином, розроблення концепції маркетингових комунікацій є ключовим етапом для формування позитивного іміджу бренду та залучення цільової аудиторії. Інтеграція традиційних і цифрових каналів забезпечує максимальну ефективність взаємодії з клієнтами.

ВИСНОВКИ

У магістерській роботі було проведено комплексне дослідження методів шифрування та їх криптографічної стійкості, що дозволило оцінити сучасний рівень захисту інформаційних систем. Було розглянуто декілька алгоритмів шифрування, а саме: шифр Цезаря, блоковий шифр, шифр Віженера, DES, AES, RC5 і метод Вернама, кожен з яких має свої переваги, недоліки та сфери застосування.

Результати дослідження продемонстрували, що надійність шифрування залежить від багатьох факторів, таких як довжина ключа, структура алгоритму, стійкість до кібер атак та правильність реалізації. Шифр Цезаря виявився найбільш вразливим, що підтвердило його непридатність для використання в сучасних умовах. Алгоритми DES і RC5 мають середній рівень захисту, але потребують вдосконалення для протидії сучасним атакам. AES підтвердив свою репутацію як одного з найбільш надійних алгоритмів для захисту інформації, тоді як метод Вернама, будучи теоретично абсолютно стійким, потребує ідеальних умов для практичного використання.

Окрему увагу було приділено розробці середовища для випробування криптографічної стійкості алгоритмів. Це середовище забезпечує не лише шифрування та дешифрування даних, а й можливість перевірки обраних методів на уразливість до атак, що дозволяє отримати цінну інформацію для оцінки їх ефективності.

Практична значущість роботи полягає в тому, що отримані результати можуть бути використані для вдосконалення існуючих систем захисту інформації, а також для навчання студентів і фахівців у галузі криптографії. Окрім того, впроваджене програмне середовище може слугувати основою для подальших досліджень у сфері інформаційної безпеки.

Загалом, результати магістерської роботи підтвердили важливість постійного вдосконалення криптографічних алгоритмів та систем захисту даних у відповідь на зростаючі загрози кіберпростору. Розроблені методи та інструменти, а також проведений криптоаналіз роблять вагомий внесок у розвиток теорії та практики сучасної криптографії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Visual Studio Code [Електронний ресурс]. URL: <https://code.visualstudio.com/docs> (дата звернення: 24.11.2024).
2. Rule J. Dynamic HTML: The HTML Developer's Guide. Addison-Wesley Professional, 1998. 288 p.
3. Professional CSS. New York : John Wiley & Sons, Ltd., 2005.
4. Vodnik S. JavaScript. Cengage Learning, 2014. 896 p.
5. Notepad++ [Електронний ресурс]. URL: <https://notepad-plus-plus.org/> (дата звернення: 24.11.2024).
6. Kranakis E., Oorschot P. C. v. Selected Areas in Cryptography. Berlin : Springer, 2011. 112 с.
7. Nath A., Ghosh S., Mallik M.A. Key cryptography using random key generator: Proceedings of International Conference on SAM-2010, 12–15 July, 2010, Las Vegas (USA). Vol. 2. P. 239–244.
8. Wang, Ming Hu. Timing evaluation of the known cryptographic algorithms: 2009 International Conference on Computational Intelligence and Security.
9. Вербіцький О. В. Вступ до криптології. Львів : ВНТЛ, 1998. 248 с.
10. Kahn D. The Codebreakers – The Story of Secret Writing. New York : Charles Scribner's Sons, 1967. 473 с. ISBN 0-684-83130-9.
11. Wiegers K. E. Software Requirements: Practical Techniques for Gathering and Managing Requirements Throughout the Product Development Cycle. Boston : Pearson Education, 2013. 672 с.
12. Bailey T., Biessek A. Flutter for Beginners. Birmingham : Packt Publishing, 2021. 370 с.
13. Бородкіна І. А. Теорія алгоритмів : посібник для студентів вищих навчальних закладів. Київ : Юніверс, 2008. 184 с.

14. Franklin J. Beginning jQuery. Berkeley, CA : Apress, 2013. URL: <https://apress.com/> (дата звернення: 24.11.2024).
15. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. Wiley, 1996. 758 p.
16. Schneier B. Secrets and Lies: Digital Security in a Networked World. Wiley, 2000.
17. Кухаренко В. М. Сучасні засоби створення інтерактивних веб-сторінок. Київ : НТУУ «КПІ», 2017. 235 с.
18. Katz J., Lindell Y. Introduction to Modern Cryptography. Chapman and Hall/CRC, 2007. 586 p.
19. Нікітін С. В. Основи криптографії : навч. посіб. Одеса : ОНУ, 2013. 304 с.
20. Node.js [Електронний ресурс]. URL: <https://nodejs.org/> (дата звернення: 24.11.2024).
21. JavaScript.info [Електронний ресурс]. URL: <https://javascript.info/> (дата звернення: 24.11.2024).
22. ES6 Features [Електронний ресурс]. URL: <https://es6-features.org/> (дата звернення: 24.11.2024).
23. W3Schools JavaScript [Електронний ресурс]. URL: <https://www.w3schools.com/js/> (дата звернення: 24.11.2024).
24. Цуркан І. М. Основи алгоритмів та програмування : навч. посіб. Харків : ХНУ, 2015. 392 с.
25. Asynchronous JavaScript [Електронний ресурс]. URL: <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous> (дата звернення: 24.11.2024).
26. OpenSSL [Електронний ресурс]. URL: <https://www.openssl.org/> (дата звернення: 24.11.2024).

27. RSA Algorithm Explained [Електронний ресурс]. URL: <https://rsaalgorithm.org/> (дата звернення: 24.11.2024).
28. Eloquent JavaScript. A Modern Introduction to Programming. 3rd edition. New York : No Starch Press, 2018. 472 p.
29. Палій О. В., Федоренко В. Л. Криптографія: теорія і практика. Київ : КНУ ім. Т. Шевченка, 2019. 412 с.
30. JavaScript Design Patterns [Електронний ресурс]. URL: <https://addyosmani.com/resources/essentialjsdesignpatterns/book/> (дата звернення: 24.11.2024).
31. Crockford D. JavaScript: The Good Parts. O'Reilly Media, 2008.
32. Babel JavaScript Compiler [Електронний ресурс]. URL: <https://babeljs.io/> (дата звернення: 24.11.2024).
33. Ukrainian Cryptography Resources [Електронний ресурс]. URL: <https://crypto.kpi.ua/> (дата звернення: 24.11.2024).
34. HTML5 and JavaScript Web Apps. New York : Apress, 2013.
35. Web Cryptography API [Електронний ресурс]. URL: <https://w3.org/TR/WebCryptoAPI/> (дата звернення: 24.11.2024).
36. Щербakov М. І. Захист інформації : алгоритми та протоколи. Харків : ХНУРЕ, 2020. 350 с.
37. Жежнич П. Основи шифрування даних. Львів : ЛНУ, 2018.
38. Харченко О. В. Основи веб-програмування. Київ : НТУУ «КПІ», 2020. 256 с.
39. ECMAScript Language Specification [Електронний ресурс]. URL: <https://tc39.es/ecma262/> (дата звернення: 24.11.2024).
40. Руденко С. В. Основи програмування мовою JavaScript. Дніпро : ДНУ, 2016.
41. Secure Hash Algorithms [Електронний ресурс]. URL: <https://csrc.nist.gov/> (дата звернення: 24.11.2024).

42. OWASP Top 10 Security Risks [Електронний ресурс]. URL: <https://owasp.org/> (дата звернення: 24.11.2024).
43. PHP Manual [Електронний ресурс]. URL: <https://www.php.net/manual/en/> (дата звернення: 24.11.2024).
44. Modern Cryptography [Електронний ресурс]. URL: <https://cryptography.io/> (дата звернення: 24.11.2024).
45. Canvas API [Електронний ресурс]. URL: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API (дата звернення: 24.11.2024).
46. Гусев В. А. Основи інформатики та програмування. Одеса : ОНУ, 2019. 298 с.
47. Ferguson N., Schneier B. Practical Cryptography. Wiley, 2003.
48. Шифрування в веб-додатках [Електронний ресурс]. URL: <https://uk.websecurity.com.ua/> (дата звернення: 24.11.2024).
49. GitHub Docs [Електронний ресурс]. URL: <https://docs.github.com/> (дата звернення: 24.11.2024).
50. Advanced Encryption Standard (AES) [Електронний ресурс]. URL: <https://nvlpubs.nist.gov/> (дата звернення: 24.11.2024).
51. Varenikov A. V., Shapovalova S. I. System of experimental research of cryptographic protection methods: Materials of the XXI Int. Conf. on Scientific Problems of Energy Support. Kyiv : KPI, 2024. Т. 2. Р. 224–225.

ДОДАТОК А

Середовище випробування криптографічної стійкості методів
шифрування

Апробація

Київ - 2024

UDC 004.08

¹ MS's programme 1st year Varenikov A.V.

¹ Assoc.prof., cand.eng.sc. Shapovalova S.I.

<https://scholar.google.com.ua/citations?user=UgfWon8AAAAJ&hl=en>

¹ Igor Sikorsky Kyiv Polytechnic Institute

SYSTEM OF EXPERIMENTAL RESEARCH OF CRYPTOGRAPHIC PROTECTION METHODS

Statement of the problem and its relevance. Cryptography is an important component of information security. Encrypted communication on the Internet is becoming increasingly important in the face of cybercriminal threats. Cryptography research and development of new encryption methods is an urgent task [1]. To ensure the reliability of encryption systems, it is important to develop mathematical models that meet modern requirements and take into account the dynamics of cyber threats [2]. Combining practical research and theoretical development is key to ensuring effective cryptographic security in today's digital world. The problem of information protection becomes more urgent in the conditions of martial law, because if messages are transmitted without reliable protection, the information can reach the enemy, which can lead to serious consequences.

Analysis of the latest research. A comparison of the reliability and stability of existing cryptographic methods against various types of attacks is given in the article [3].

Encryption is based on mathematical models and algorithms that transform data into an unreadable form that can only be decoded by a person with the appropriate decryption keys. This ensures the confidentiality and integrity of data as it travels over the network.

Goal formulation. The purpose of the work is to evaluate the effectiveness of cryptographic protection for certain types of cryptosystems, to obtain estimates of their resistance to attacks depending on the conditions in which they function, as well as to use new methods to improve cryptographic information protection systems, taking into account various calculation models and development prospects modern directions of cryptanalysis, means of computer technology.

Main part. Modern cryptography is divided into 4 sections (Fig. 1).

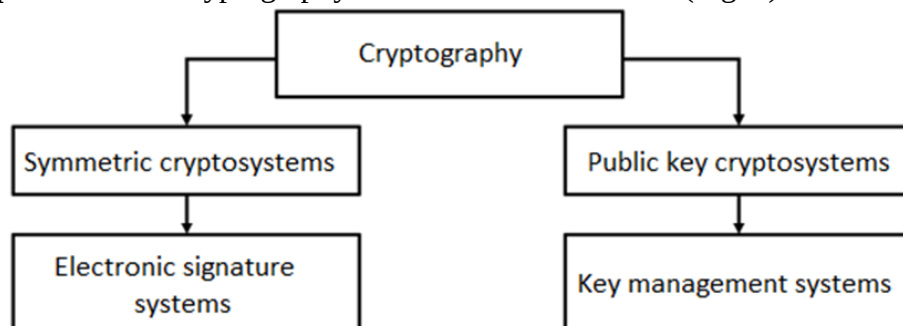


Figure 1 – Sections of modern cryptography

The main areas of use of cryptographic methods are the transmission of confidential information through communication channels (for example, e-mail), establishing the authenticity of the transmission of messages, storing information (documents, databases) on media in an encrypted form. Cryptography methods are divided into two classes: 1) information processing by replacing

and moving letters, in which the amount of data does not change (encryption); 2) compression of information through the replacement of selective combinations of letters, phrases, or words (encoding).

Cryptographic methods can be implemented in software and hardware [4]. To protect text information during its transmission over long distances through telecommunication stations, cryptographic methods implemented in hardware are used.

Both hardware and software methods are possible for exchanging information between computers over a telecommunications network, as well as for working with local subscribers. Software methods of encryption and coding are used to store information on magnetic media. Hardware methods of information encryption are used to transmit protected data over a telecommunications network [5].

Existing cryptographic methods differ in their reliability, complexity of implementation, type of encryption, etc. A comparative analysis is given in [3] (Table 1).

Table 1 – Comparative analysis of existing cryptographic methods [3]

Algorithm	Key size	Block size	Round	Structure	Flexible	Features
DES	64 bits	64 bits	16	Festiel	No	Not strong enough
DH	Variable	-	-	Public key	Yes	Good security and low speed
E-DES	1024 bits	128 bits	16	Festiel	-	Good security and fast speed
RSA	1024 to 4096	128 bits	1	Public key	No	Excellent security and low speed
T-DES	112 or 168	64 bits	48	Festiel	Yes	Adequate security and fast
ECC	More than symmetric and variable	Variable	1	Public key	Yes	Excellent security and fast speed
EEE	1024 bits	-	-	Public key	Yes	Enough secured and fast speed
RC4	Variable	40-2048	256	Festiel Stream	Yes	Fast cipher
RC2	8,128,64 bits	64 bits	16	Festiel	-	Good and fast security

Thus: 1) the most reliable and fast is the ECC encryption method; 2) the least reliable and least fast is the DH encryption method; 3) the longest encryption key contains the RSA encryption method.

Conclusions. The basic methods of encryption are defined for conducting computational experiments to determine the optimal cryptographic protection.

References:

1. Kranakis E., Oorschot P. C. v. Selected Areas in Cryptography : навч.-метод. посібник. Берлін: Springer, 2011. 112 с.
2. A.Nath, S.Ghosh, M.A.Mallik «Key cryptography using random key generator»: Proceedings of International conference on SAM-2010 held at Las Vegas(USA) 12-15 July, 2010,

Vol-2, P-239-244.

3. Abood O. G., Guirguis S. K. A Survey on Cryptography Algorithms. International Journal of Scientific and Research Publications (IJSRP). 2018. Vol. 8, no. 7. URL: <https://doi.org/10.29322/ijsrp.8.7.2018.p7978> (date of access: 13.03.2024).

4. Verbickiy O. V. Vstup do kryptologiyi. – L. : VNTL, 1998. – 248 p.

5. Neal Koblitz, «A Course in Number Theory and Cryptography»: Second Edition Published by Springer, Vol-4,P-87-94.