МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітньо-професійна програма: «Безпека комп'ютерних систем і мереж»

Група: 4КБ-02

Дипломний проект

здобувача освіти денної форми навчання
КБ.02.04.000.ДП

БОДАЧЕВСЬКОГО
ДАНІЇЛА ВАСИЛЬОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: «**Безпека комп'ютерних систем і мереж»**

Група: **4КБ-02**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Моделювання роботи кодеку модифікованого криптоалгоритму DES

Проектний матеріал складається з пояснювальної записки на 76 сторінках та графічного (презентаційного) матеріалу на 16 аркушах (слайдах)

Дипломник _____ (Бодачевський Д.В.)

Керівник _____ (Кривченко Ю.В.)

Консультанти:

з економічного розділу _____ (Канський М.Ю.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н.І.)

з нормоконтролю _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Краснокутська К.Г.)

Захист «26» червня 2025 р. Протокол ЕК № 5

Оцінка ЕК 5 (відмінно) / 97 б.

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Безпека комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР *Беркань І.В.*

"19" 06 2025 р.

ЗАВДАННЯ

на дипломний проект

Здобувачеві освіти *Бодачевському Данііу Васильовичу*

(прізвище, ім'я, по батькові)

1. Тема проекту *Моделювання роботи кодеку модифікованого криптоалгоритму DES*

затверджена наказом по коледжу від "14" 11 2024 р. № 246

2. Термін здачі закінченого проекту 16.05.25

3. Вихідні данні до проекту *1. Реалізувати модель криптостійкого кодеку на базі симетричного блокового шифру - одного з модифікованих алгоритмів DES; 2. Передбачити шифрування і дешифрування текстових повідомлень з файлів .txt; 3. Реалізувати візуальний інтерфейс користувача для створюваного додатку; 4. Передбачити перевірку роботи кодеку за допомогою моделювання у програмному середовищі Cryptool2*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Аналітичний огляд методів і засобів симетричної криптографії; Визначення структури і властивостей криптоалгоритму DES; Аналіз існуючих модифікацій криптоалгоритму DES; Моделювання роботи алгоритму модифікованого DES у CrypTool2; Підготовка алгоритмів для реалізації кодека модифікованого DES; Підготовка засобів розробки програмного кодеку; Реалізація програмного застосунку кодеку криптоалгоритму модифікованого DES

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Загальна структурна схема криптосистеми; Схема блокового шифрування у криптосистемі; Базова структура мережі Фейстеля; Блок-схема алгоритму шифрування DES та генерування ключів; Принципи шифрування модифікованими криптоалгоритмами DES; Результати моделювання у CrypTool2; Структурна схема роботи модифікованого криптоалгоритму DES; Блок-схема алгоритму дій користувача; Інтерфейс програми-кодеку на базі модифікованого алгоритму DES

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Кривченко Ю.В.*		
Економічний розділ	*Канський М.Ю.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання 15.05.25.

Керівник *Кривченко Ю.В.* _____ (підпис)

Завдання прийняв до виконання *Бодачевський Д.В.* _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	*Вступ. Постановка задачі проектування*	15.05.25	виконано
2	*Аналіз технічного завдання та пошук літератури*	15.05.25	виконано
3	*Аналіз властивостей алгоритмів симетричного*		
	шифрування	16.05.25	виконано
4	*Порівняльний огляд сучасних алгоритмів шифрування*	17.05.25	виконано
5	*Визначення типів блокових алгоритмів шифрування*	19.05.25	виконано
6	*Визначення структури і властивостей DES;*	06.06.25	виконано
7	*Аналіз існуючих модифікацій криптоалгоритму DES*	07.06.25	виконано
8	*Моделювання роботи алгоритму модифікованого DES у*		
	CrypTool2	08.06.25	виконано
9	*Розробка моделі кодеку та його програмна реалізація*	09.06.25	виконано
10	*Реалізація візуального інтерфейсу застосунку*	10.06.25	виконано
11	*Написання коду програми мовою C#*	11.06.25	виконано
12	*Випробування застосунку та аналіз результатів*	12.06.25	виконано
13	*Виконання економічних розрахунків*	13.06.25	виконано
14	*Розробка питань з охорони праці та техніки безпеки*	14.06.25	виконано
14	*Підготовка мультимедійної презентації проекту*	13.06.25	виконано

Дипломник _____ (підпис)

Керівник _____ (підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КБ 02. 04 000. 00 ДП	Дипломний проект		
А4			КБ 02. 04 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КБ 02. 04 000. 00 ДП		
Розробив		Бодачевський Д.		16.06	Моделювання роботи кодеку модифікованого криптоалгоритму DES	Літ.	Аркуш 4 / Аркушів 76
Перевірив		Кривченко Ю.В.				Н Д П	
Н. контр.		Петрашова В.І.		16.06.15		ВСП "ОТФК ОНТУ" гр.4КБ-02	
Затверд.		Кривченко Ю.В.		16.06			

ЗМІСТ

ВСТУП

Нині питання інформаційної безпеки та захисту даних набувають критичного значення в умовах стрімкого розвитку комп'ютерних технологій та зростання кіберзагроз. Дипломний проект, присвячений моделюванню роботи кодеку модифікованого криптоалгоритму DES, обґрунтовано потребою вдосконалення класичних алгоритмів шифрування, адаптуючи їх до сучасних вимог щодо надійності та ефективності захисту інформації.

Алгоритм DES, який був стандартом у сфері симетричного шифрування протягом тривалого часу, незважаючи на виявлені недоліки у зв'язку з еволюцією обчислювальних потужностей, залишається важливим прикладом історичного розвитку криптографії. Модифікація DES з метою усунення слабких місць та підвищення пристосованості алгоритму до сучасних завдань є актуальним напрямком досліджень. У даному проекті розглядається концепція побудови спеціалізованого кодеку, здатного здійснювати як шифрування, так і дешифрування даних із застосуванням змінених параметрів класичного алгоритму.

Метою роботи є розробка та аналіз математичної та програмної моделі роботи кодеку, що базується на модифікованій версії DES, а також визначення основних характеристик його ефективності та стійкості до криптоаналізу. Завдання дослідження включають:

- Аналіз теоретичних основ алгоритму DES та його модифікацій;

- Розробку логічної схеми та програмної реалізації кодеку;

- Проведення моделювання роботи кодеку з метою оцінки його продуктивності та надійності;

- Порівняльний аналіз отриманих результатів з класичними підходами до організації криптографічних систем.

Особлива увага приділяється практичній значущості дослідження, адже створена модель може стати базою для подальшого вдосконалення систем забезпечення інформаційної безпеки. Результати роботи можуть бути використані як в академічних дослідженнях, так і у впровадженні реальних проектів, що вимагають високого рівня захисту даних.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Аналітичний огляд методів і засобів симетричної криптографії

Нині симетрична криптографія займає центральне місце у сфері інформаційної безпеки, адже забезпечує швидке та ефективне шифрування даних за допомогою одного спільного ключа для процесів як шифрування, так і дешифрування. Завдяки своїй простоті та високій швидкодії методи симетричного криптографування стали наріжним камінням численних систем захисту інформації, що знаходять застосування як у комерційних, так і в державних структурах.

Аналітичний огляд методів симетричної криптографії починається з розгляду базових принципів, на яких спирається алгоритмічна база сучасних криптосистем. Найпоширенішими серед них є блокові та потокові шифри. Блокові криптоалгоритми, зокрема класичний DES, який піддався численним модифікаціям задля підвищення стійкості до криптоаналізу, демонструють принцип розбиття даних на фіксовані блоки з подальшою їх обробкою згідно з заданим алгоритмом. Такий підхід дозволяє комплексно розглянути механізми заміщення й перестановки, що слугують основою для досягнення високої ступені заплутування інформації.

Особливості блокових криптоалгоритмів проявляються через їх внутрішню структуру: розбиття даних, роботу з ключами через визначені раунди обробки, застосування специфічних S-блоків та перестановок. Ці елементи є основою не лише для визначення криптографічної стійкості самої системи, але й для подальшого аналізу можливих векторів атак. Сучасне дослідження в області симетричного шифрування спрямовано на вдосконалення цих механізмів з метою підвищення якості захисту, що знаходить відображення і у розробці модифікованого варіанту алгоритму DES.

Крім базової структури, системи симетричної криптографії враховують додаткові аспекти, зокрема забезпечення цілісності даних, застосування хеш-функцій та використання генераторів випадкових чисел. Хеш-функції слугують для підтвердження незмінності зашифрованих даних та виявлення можливих змін після передачі, що набуває особливої актуальності у сучасних умовах активної передачі

інформації по мережах. Використання генераторів випадкових чисел допомагає у створенні надійних криптографічних ключів і, як наслідок, запобігає виникненню передбачуваних закономірностей у процесі шифрування, що є важливим чинником у запобіганні частотному аналізу.

У наступних підрозділах зазначені теми розглядаються більш детально із врахуванням специфіки розробки моделі кодеку модифікованого криптоалгоритму DES, що може бути реалізовано, зокрема, із застосуванням засобів криптографічного пакету Cryptotool2.

1.1.1 Аналіз організації кодеків симетричних криптоалгоритмів

Сучасні засоби криптографічного захисту включають широкий спектр алгоритмічних рішень, які, залежно від способу застосування ключів, поділяються на методи з єдиним секретним параметром (симетричні) та ті, що функціонують на основі парних ключів (асиметричні). У випадку симетричних систем особлива увага приділяється саме кодекам — інтегрованим модулям, що об'єднують в собі процеси шифрування і декодування даних за допомогою одного конфіденційного параметру.

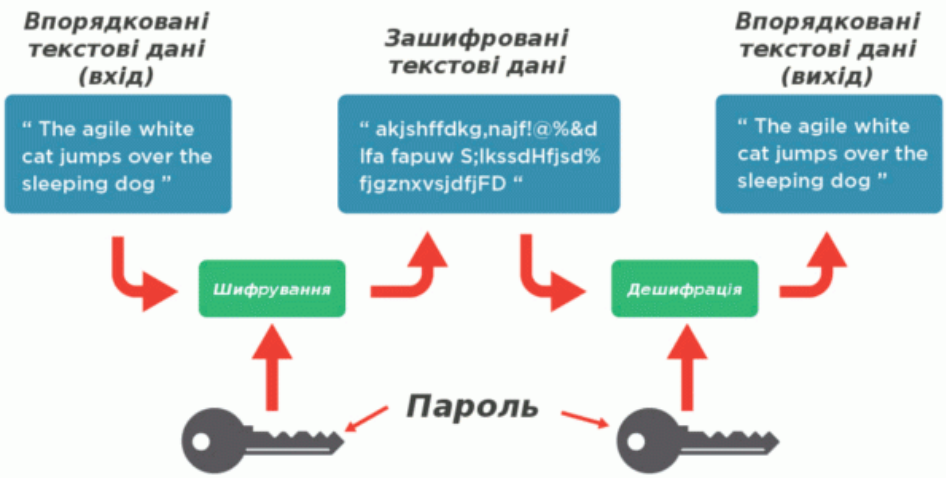


Рисунок 1.1. Принцип симетричного шифрування у криптосистемі

На рис.1.1 показано інтегровану схему симетричного кодування. Тут видно, як відкритий текст перетворюється у захищену форму, а потім відновлюється за допомогою одного спеціально згенерованого параметра. Цей ключ, який формується за допомогою різних алгоритмічних методів для забезпечення його унікальності (як випадкова бінарна послідовність або специфічний пароль), є ядром всієї криптографічної системи. Оскільки базові алгоритмічні принципи

перебувають у відкритому доступі, компрометація ключа може призвести до негайного розкриття зашифрованої інформації, що підкреслює важливість його надійного управління.

Організація кодеків симетричних криптоалгоритмів базується на двох основних підходах до обробки вхідних даних — потоковому та блочному шифруванні.

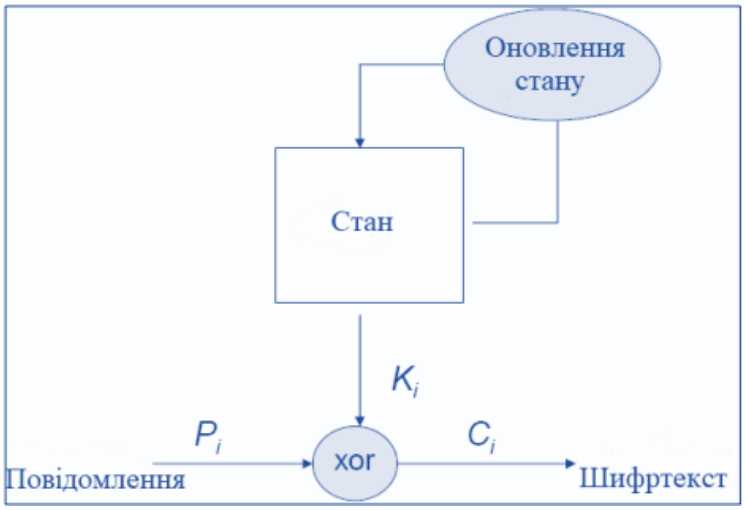


Рисунок 1.2. Схема потокового шифрування у криптосистемі

На рис. 1.2 показано схему послідовного перетворення даних, продемонстровано, як потокові методи обробляють інформацію на рівні окремих бітів. Цей спосіб забезпечує максимально оперативне кодування, оскільки дані не потребують попереднього накопичення для формування блоків, що дуже важливо в апаратних рішеннях та у протоколах захищеного зв’язку.

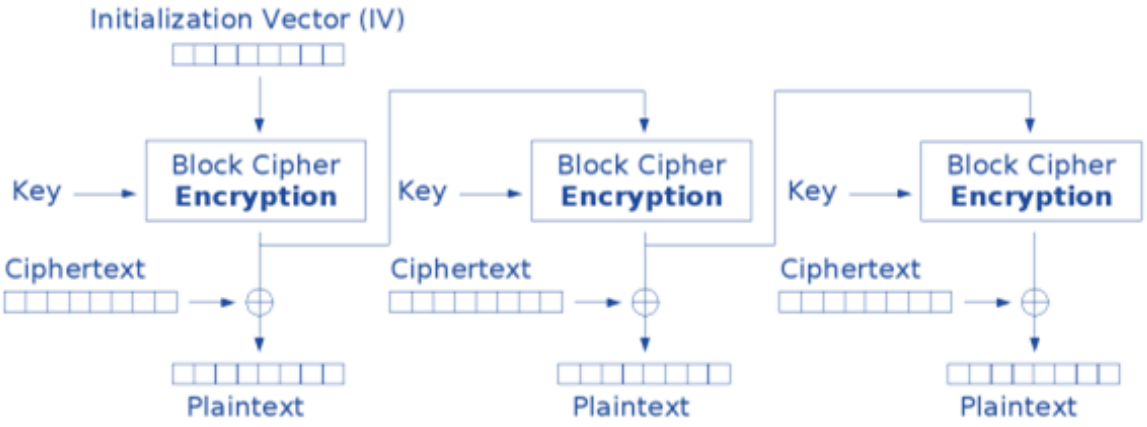


Рисунок 1.3. Схема блокового шифрування у криптосистемі

Альтернативним підходом є блочне шифрування, яке наочно представлено на рис. 1.3. У цьому випадку вхідний текст розбивається на сегменти фіксованого

розміру, кожен з яких зазнає серії трансформацій, включаючи заміну, перестановку та побітові операції. Завдяки цьому досягаються високі рівні дифузії та заплутування, що значно ускладнює спроби криптоаналізу за відсутності правильного ключа.

Особливе місце в архітектурі кодеків займає модуль управління ключами, який відповідає за перетворення початкового секретного параметру в набір підключів для послідовних етапів процесів шифрування і дешифрування. Цей процес, наочно зображений на рис. 1.4, підвищує стійкість системи. Завдяки адаптації класичних алгоритмів, таких як модифікована версія DES, до сучасних вимог безпеки, можливе посилення захисту даних через варіативність внутрішньої структури (наприклад, шляхом заміни стандартних блоків підстановок та перестановок).

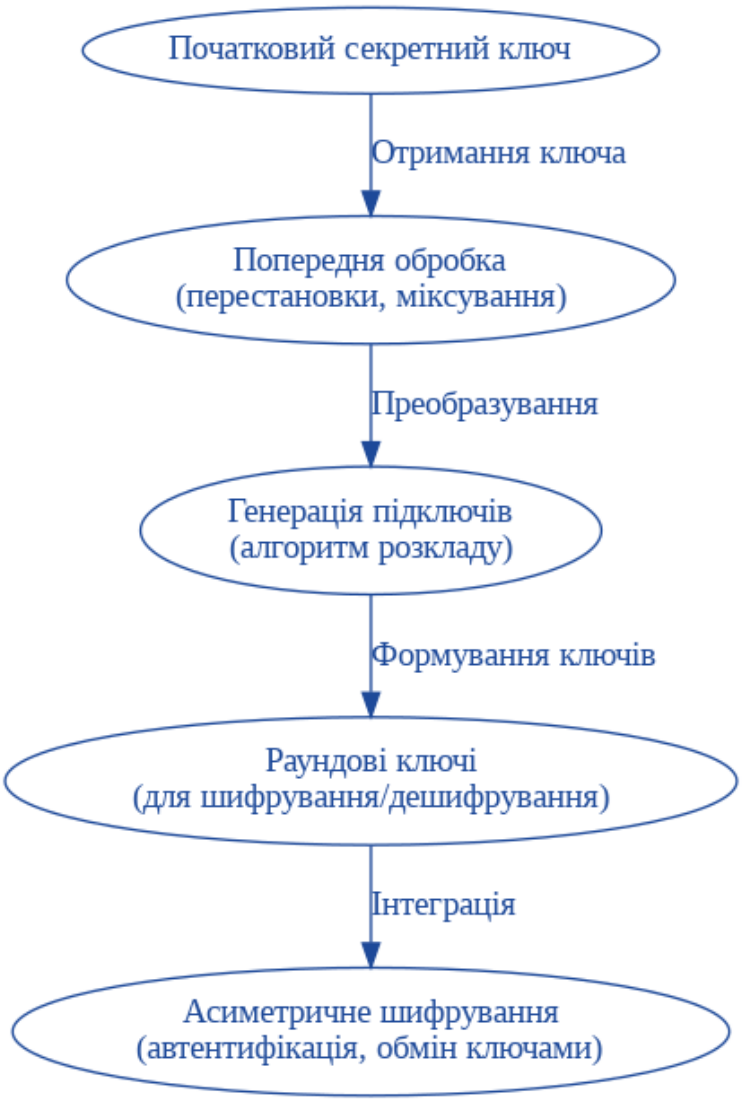


Рисунок 1.4. Модуль генерування та трансформації ключів

Крім основних компонентів, у вдосконалених рішеннях часто використовують додаткові візуальні схеми для представлення процесу інтеграції симетричних і асиметричних методів. Наприклад, рис. 1.5 ілюструє, як симетричне шифрування може взаємодіяти з асиметричними технологіями для забезпечення автентифікації та додаткового рівня захисту. Такий підхід дозволяє не лише зберегти оперативність операцій, але й значно знизити ризики, пов’язані з передачею секретних параметрів.

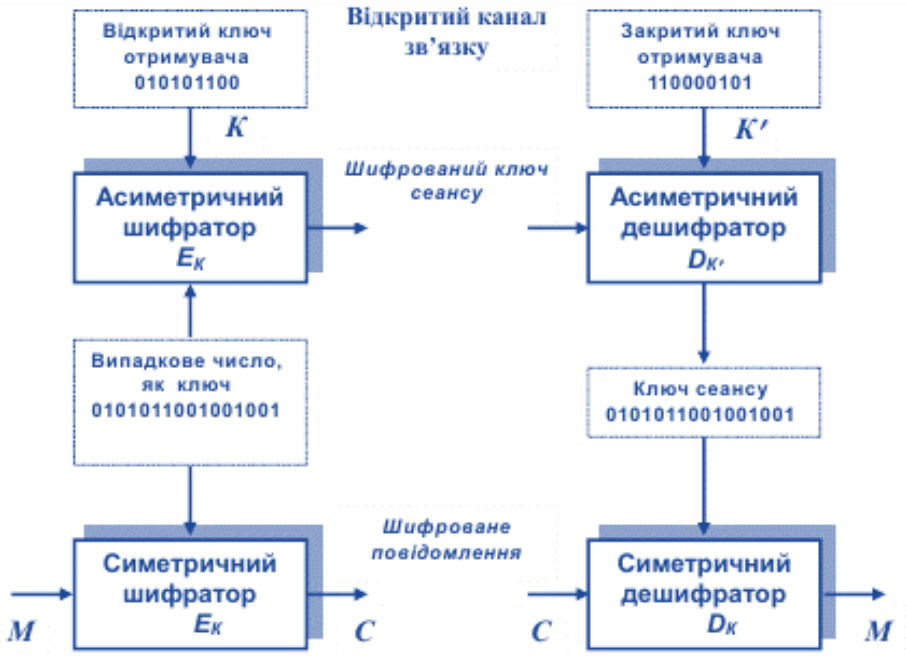


Рисунок 1.5. Комбінована система розподілу ключів

Організація кодеків симетричних криптоалгоритмів базується на багаторівневій модульній структурі, що поєднує адаптивне управління ключами, початкове перемішування даних та застосування різноманітних методів (як потокових, так і блочних) шифрування. Така інтеграція дозволяє створювати високоефективні та стійкі криптографічні системи, які відповідають сучасним вимогам безпеки й можуть стати надійною платформою для подальших досліджень і практичного впровадження, зокрема, із застосуванням таких інструментів, як Cryptotool2.

1.1.2 Аналіз принципів роботи блокових криптосистем

Блокові криптосистеми працюють за принципом обробки потоків інформації, що розбиті на невеликі сегменти фіксованої довжини. Кожен такий сегмент (блок) перетворюється за формулами

$$Z = EnCrypt(X, Key) \quad (1.1)$$

$$X = DeCrypt(Z, Key), \quad (1.2)$$

де Key — секретний параметр, який забезпечує збереження конфіденційності. Варто відзначити, що вихідний блок X і шифротекст Z мають однакову розрядність, хоча ця величина не обов'язково співпадає з довжиною ключа.

Блокові шифри забезпечують перетворення даних таким чином, що навіть при наявності початкового та зашифрованого блоків відновити ключ можна лише шляхом повного перебору. Це є невід'ємною умовою високої криптостійкості: при відомих деяких стандартних вставках чи заголовках файлів зловмиснику доступний лише один метод — перебір усіх можливих варіантів ключа. Крім того, методологія побудови блокових систем дозволяє створювати ланцюжки з окремих байтових одиниць, що шифруються за допомогою алгоритмів із фіксованою обробкою, забезпечуючи тим самим можливість кодування інформації довільної довжини.

У схемі блокового шифрування (рис.1.3) вхідний блок даних перетворюється за допомогою функції шифрування, що приймає два аргументи — сам текст та ключ. Вихідний блок отримує фіксовану розрядність, що гарантує уніфікованість обробки. Давши змогу працювати із даними будь-якого об’єму, блокові шифри є фундаментом для побудови більшості сучасних криптографічних систем.

Таблиця 1.1. Огляд сучасних симетричних криптоалгоритмів

Назва алгоритму	Розмір блоку	Довжина ключа
AES	*128 біт*	*128, 192 або 256 біт*
IDEA	*64 біт*	*128 біт*
CAST128	*64 біт*	*128 біт*
BlowFish	*64 біт*	*від 128 до 448 біт*
ГОСТ	*64 біт*	*256 біт*
TwoFish	*128 біт*	*від 128 до 256 біт*
MARS	*128 біт*	*від 128 до 1048 біт*

У сучасних криптосистемах блокове шифрування застосовується не лише для обробки даних фіксованої довжини, а й для створення ланцюжків зашифрованих байтів, що дозволяє ефективно кодувати інформацію довільної величини. Це викладено у табл. 1.1, де наведено основні характеристики алгоритмів, що

використовують блокове шифрування. Зокрема, сучасні системи, як-от AES, IDEA чи BlowFish, демонструють різні параметри розміру блоку та довжини ключа.

Для детальнішого аналізу операцій, що виконуються над блоками, варто звернути увагу на перелік математичних перетворень, використовуваних у процесі шифрування. Як показано у табл. 1.2, блочне шифрування здійснюється за допомогою операцій додавання, операції XOR, множення за модулем, бітових зсувів та табличних підстановок. Ці операції можуть комбінуватися в залежності від конкретної реалізації: параметр V у формулах може бути постійним числом, функцією від ключа чи обчисленим значенням із частини самого блоку.

Таблиця 1.2. Базові бінарні операції у блокових криптоалгоритмах

Операція	Формула перетворення
Додавання	$X' = X + V$
Виключне АБО (XOR)	$X' = X \; XOR \; V$
Множення за модулем $(2^N + 1)$	$X' = (X \cdot V) \; mod \; (2^N + 1)$
Множення за модулем (2^N)	$X' = (X \cdot V) \; mod \; (2^N)$
Арифметичний зсув (вліво/вправо)	$X' = X \; SHL \; V \; / \; X' = X \; SHR \; V$
Циклічний зсув (вліво/вправо)	$X' = X \; ROL \; V \; / \; X' = X \; ROR \; V$
Таблична підстановка (S-box)	$X' = Table[X, V]$

В кожній із цих операцій параметр V може виступати у різних ролях:

- як постійне значення (наприклад, X' = X + 125);
- як результат функції від ключа (наприклад, X' = X + F(Key));
- або як значення, обчислене з частини самого блоку (наприклад, $X_2' = X_2 + F(X_1)$).

Для забезпечення стійкості системи, де відомими можуть бути як початковий текст, так і зашифрований блок, сучасні шифри розраховані таким чином, що підбір ключа можливий лише шляхом повного перебору всіх варіантів. Це вимагає високої складності використання ключа, що стає практичним завданням криптографічних розробників. Однією з популярних архітектур, яка задовольняє ці вимоги, є мережа Фейстеля. Суть класичної Фейстелевої мережі полягає в наступних кроках:

- Розбиття блоку: Вхідний блок ділиться на дві рівні частини – ліву (L) та праву (R).

- Обчислення функції: Функція f(R, K), залежна від раундового ключа K, обробляє праву частину.

- Комбінування: Результат функції об'єднується з лівою частиною за допомогою операції, наприклад, XOR.

- Обмін: Значення правої частини передаються в ліву і використовуються для наступного раунду.

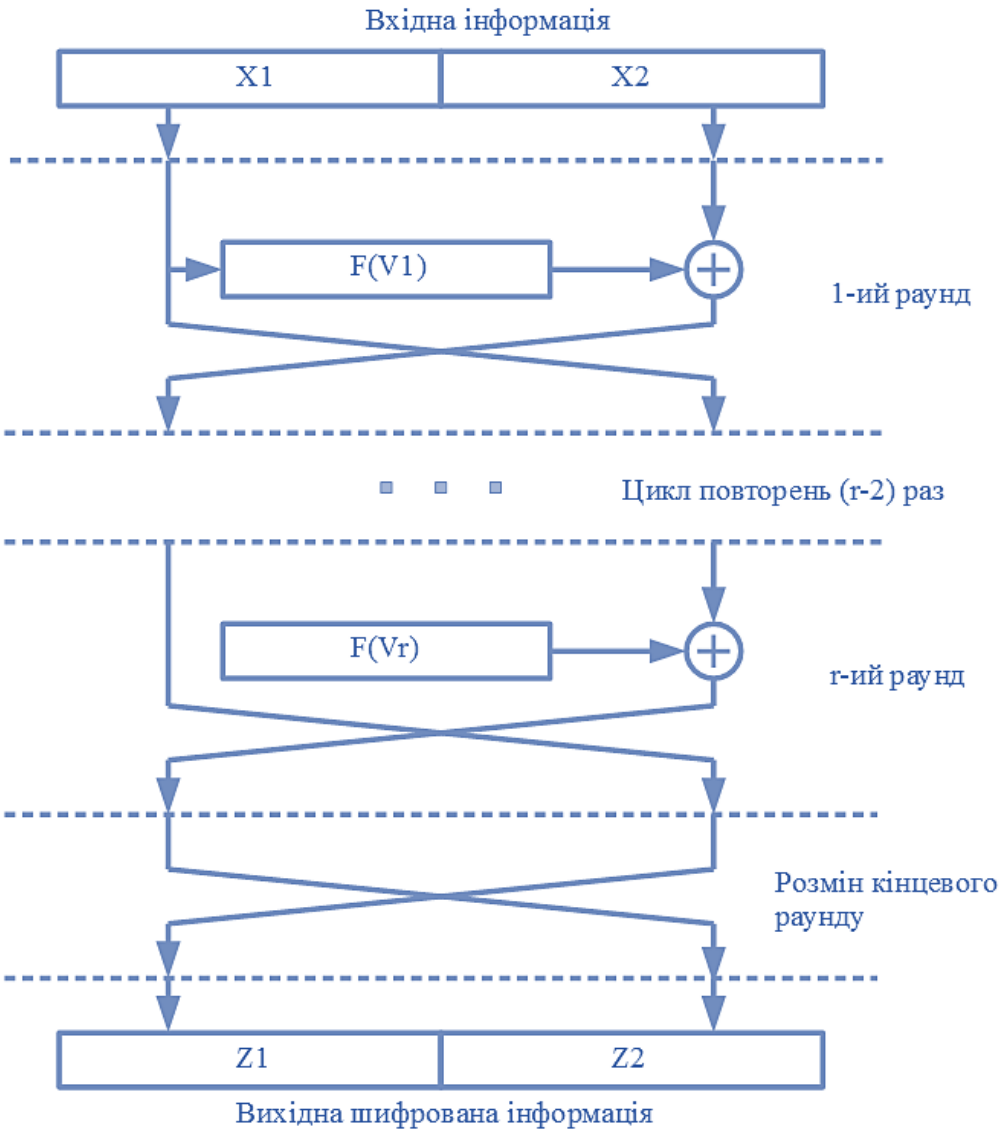


Рисунок 1.6. Базова структура мережі Фейстеля при шифруванні

Схема, зображена на рис. 1.6, гарантує оборотність системи: для дешифрування необхідно лише застосувати зворотні операції з відповідною послідовністю раундових ключів. Завдяки цій моделі, навіть при використанні необоротних перетворень у функції f, загальна операція залишається відновлюваною.

Для роботи з великими блоками даних (128 біт і більше) розробники застосовують модифікації класичної мережі Фейстеля. Сучасні рішення — конфігурації тип-2 та тип-3 — ілюструються на рис.1.7 (модернізована Фейстелева мережа, Конфігурація тип-2, модернізована Фейстелева мережа, Конфігурація тип-3). Такі розширення дозволяють підвищити швидкість обробки та рівень перемішування даних, що є критичним при роботі з високорозрядними блоками.

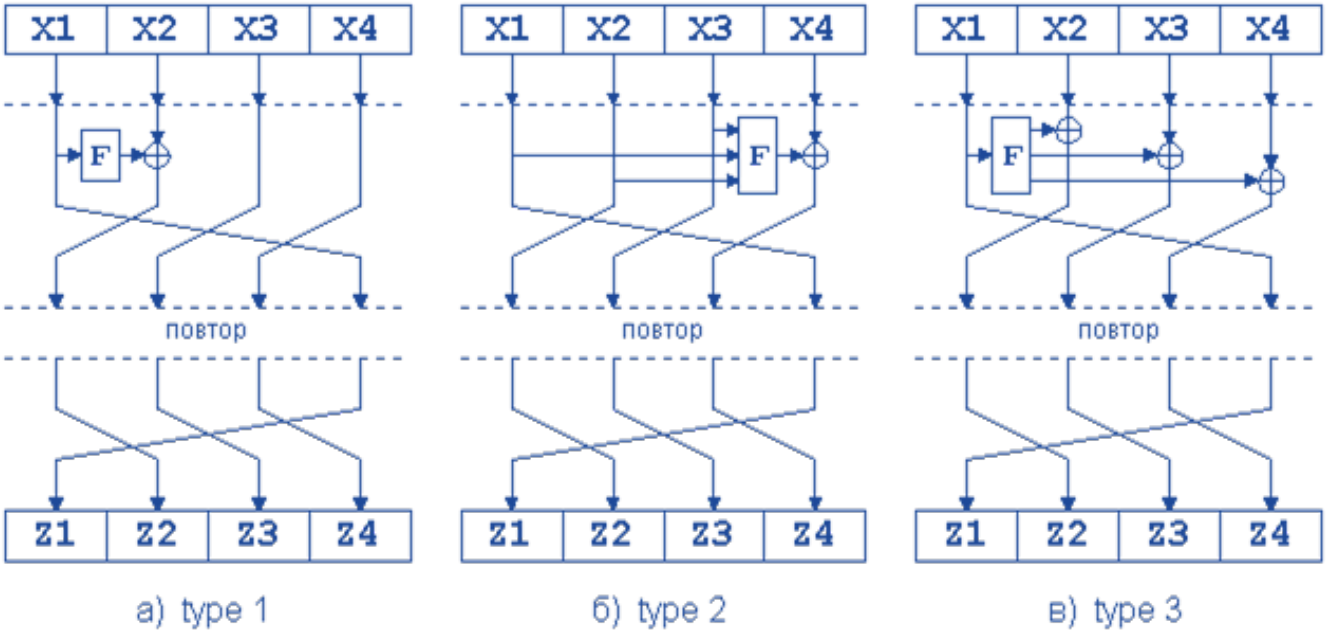


Рисунок 1.7. Варіанти модифікації мережі Фейстеля при шифруванні

Окрім описаних операцій, широко застосовується також технологія розширення ключа (key scheduling), що полягає у попередньому обчисленні проміжних значень функцій від ключа. Це сприяє збільшенню продуктивності без зміни основних криптографічних характеристик алгоритму.

Блокові криптосистеми здійснюють перетворення даних за допомогою послідовних операцій, що забезпечують високий рівень дифузії та заплутування. Як показано за допомогою функцій EnCrypt та DeCrypt у поєднанні з переліком бінарних операцій (табл. 1.2), ефективна робота шифрування забезпечується суворим підбором ключів.

Схема мережі Фейстеля, яка описана вище, дозволяє одночасно гарантувати оборотність і стійкість системи до криптоаналізу. Сучасні рішення, наведені у табл. 1.1, підтверджують актуальність цих підходів для побудови надійних криптографічних систем у цифрову епоху.

1.1.3 Аналіз засобів генерування та трансформація ключів

Ефективність симетричних криптосистем у великій мірі залежить від надійності процесів генерування та подальшої трансформації ключів. Сам ключ, як центральний параметр для шифрування та дешифрування, має бути не лише випадковим і стійким до перебору, а й проходити комплексне перетворення (key scheduling) для отримання набору підключів, які використовуються протягом усіх раундів операцій. Ці процеси не лише забезпечують високий рівень дифузії і заплутування всередині криптосистеми, але й гарантують, що відновити первинний ключ з раундових підключів за відомих вхідних та вихідних даних практично неможливо.

Основним підходом до генерування ключів є використання криптографічно стійких генераторів випадкових чисел або застосування функцій добування ключа (Key Derivation Functions, KDF), що дозволяють отримати необхідну бінарну послідовність із заданих вхідних даних або паролів. Отриманий початковий ключ підлягає додатковій трансформації за допомогою спеціалізованих алгоритмів перетворення, що включає застосування перекладень бітів, перестановок, операцій додавання, бітових зсувів та таблиць підстановок. Ці математичні операції й їх комбінації забезпечують високий ступінь нелінійності та перемішування, що критично для стійкості криптосистеми до атаки за відомими текстами.

Змінна, значення якої не можна передбачити, являє собою випадкову величину. Послідовність таких чисел характеризується максимальною непередбачуваністю, що визначається абсолютною інформаційною ентропією. Нехай x – випадкова величина, яка може приймати значення x_i з ймовірностями p_i. Тоді інформаційну ентропію можна визначити за наступною формулою:

$$H(X) = -\sum_i p_i \log_2 p_i \qquad (1.3)$$

Ця оцінка демонструє, що при рівномірному розподілі ймовірностей ентропія максимально висока, що ідеально підходить для криптографії. Випадкові величини поділяються на дискретні та неперервні. До дискретних величин належать ті, де:

- ймовірності кожного можливого значення знаходяться в інтервалі [0,1] і їх сума дорівнює 1;

- кількість можливих значень є скінченною або нескінченно зліченною;
- дані отримуються шляхом підрахунку.

Неперервні величини можуть приймати будь-яке значення на певному інтервалі, їх розподіл описується кривою щільності, а кількість можливих значень практично нескінченна, що отримується через процес вимірювання.

Математичне сподівання є ще одним ключовим поняттям у теорії ймовірностей. Для дискретної випадкової величини воно визначається як:

$$E[X] = \sum_i x_i \, , p_i \tag{1.4}$$

Для неперервної величини – як інтеграл від x по функції густини $f(x)$:

$$E[X] = \int_{-\infty}^{+\infty} x \, , f(x), dx \tag{1.5}$$

Ідеальний генератор випадкових чисел, який видає значення на інтервалі (0,1) з рівномірним розподілом, гарантує, що кожен результат має однакову ймовірність. Однак комп'ютери використовують алгоритмічні методи для формування послідовностей випадкових чисел, що робить їх псевдовипадковими або квазі-випадковими, тобто мають нижчу ентропію через залежність від заданого початкового значення *seed*.

Для оцінки якості випадковості застосовують поняття Колмогоровської складності, яка є мірою мінімальної довжини алгоритмічного опису деякого рядка даних. Формально, Колмогоровську складність можна записати як:

$$K(s) = \min, |p|, :, U(p) = s \tag{1.6}$$

де s – заданий рядок, $|p|$ – довжина програми, а U – універсальна машина Тюрінга. На практиці обчислити $K(s)$ неможливо через відсутність ефективних алгоритмів, що робить цю міру корисною переважно як теоретичну.

Також широко застосовується метод Монте-Карло, який, базуючись на законі великих чисел, дозволяє через численні повтори оцінювати ймовірнісні характеристики систем, що є важливим при аналізі незалежності отриманих випадкових величин.

Для забезпечення високої криптографічної стійкості важливо, щоб ключі формувалися із максимальним рівнем ентропії. Саме тому в криптографії використовуються криптографічно стійкі генератори випадкових чисел

(Cryptographically Secure Pseudorandom Number Generator, CSPRNG), які, приймаючи на вхід початкове значення (*seed*), генерують послідовності, близькі до ідеально випадкових.

Схема, зображена на рис. 1.4, демонструє основні етапи роботи:

1. Початкове формування ключа. На цьому етапі генерується первинний секретний параметр за допомогою криптографічного генератора випадкових чисел або на базі KDF;

2. Попередня обробка. Первинний ключ піддається початковому перемішуванню (наприклад, за допомогою перестановок і міксування) задля посилення його властивостей;

3. Ключове розкладання (key scheduling). За допомогою спеціальних функцій первинний ключ трансформується у набір раундових підключів, що використовуються протягом процесу шифрування;

4. Кешування проміжних результатів. При багаторічному застосуванні одних і тих же ключових параметрів для численних операцій відбувається оптимізація – проміжні обчислення зберігаються, що дозволяє підвищити швидкості роботи без зниження криптографічних характеристик.

Для детальнішого аналізу операцій, що використовуються у трансформації ключів, у табл. 1.3 наведено перелік основних бінарних операцій, які застосовуються для перетворення первинного ключа в залежності від конструкції алгоритму.

Таблиця 1.3. Основні операції в трансформації ключів

Операція	Опис
Побітове виключне АБО	*Забезпечує перемішування бітів шляхом комбінування початкового ключа із спеціальним вектором*
Додавання за модулем	*Використовується для зсуву значень ключа, що забезпечує непередбачуваність результату*
Бітові зсуви (ротації)	*Ротація або зсув бітового представлення ключа для посилення дифузії*
Табличні підстановки	*Застосування S-блоків для нелінійних перетворень, що змішують вхідні дані ключа*

Застосування таких операцій дозволяє досягти двох ключових властивостей:

- Оборотність інтерфейсу шифрування. Хоча сам алгоритм шифрування є оборотним для розшифрування повідомлень, трансформації ключа повинні бути побудовані таким чином, щоб процедура розкладання ключа була стійкою до зворотного аналізу;
- Невідновлюваність ключа. Навіть при розкритті раундових підключів з добрим рівнем перемішування, первинного ключа відновити не можливо без проведення повного перебору.

Ключове розкладання є критичним елементом сучасних блокових криптосистем, оскільки численні повторення операцій над одним і тим же ключем можуть стати перевантаженням для системи без попереднього кешування проміжних результатів. Тому ефективна реалізація key scheduling дозволяє не лише зекономити обчислювальні ресурси, а й зберегти криптографічну стійкість алгоритму.

1.1.4 Застосування операції XOR у симетричній криптографії

Операція виключного АБО (XOR) є однією з базових і найважливіших бінарних операцій у симетричних криптографічних системах. Її фундаментальна властивість полягає в тому, що вона є самовідновлюваною: для будь-якого біту A і ключового біту B справджуються співвідношення

$$A \; XOR \; B \; XOR \; B = A \qquad (1.7)$$

Ця властивість робить операцію XOR ідеальною для як шифрування, так і дешифрування, оскільки застосування одного й того самого ключа дозволяє відновити вихідний текст. Саме тому алгоритми типу потокових шифрів часто реалізують процес за допомогою операції XOR: генерують криптографічно стійкий потік бітів (keystream), який потім поелементно комбінують із відкритим текстом за правилом

$$X' = X \; XOR \; V, \qquad (1.8)$$

де X – відкритий текст, а V – бітовий потік або ключова маска.

Схема, зображена на рис. 1.8, демонструє простоту та ефективність застосування операції XOR у криптографічних перетвореннях. Завдяки її асоціативності та комутативності, XOR використовується не лише як прямий спосіб

шифрування, але й як базовий будівельний блок у більш складних алгоритмах, наприклад, для реалізації механізмів "whitening" або в рамках мереж Фейстеля, де операція XOR виступає у ролі інвертованої функції для забезпечення оборотності.

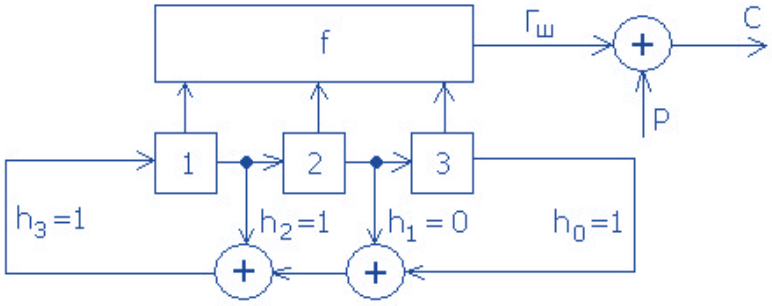


Рисунок 1.8. Застосування операції XOR у шифруванні

Окрім того, операція XOR має низький обчислювальний ресурсоспоживання, що дозволяє реалізувати її ефективно як у програмному забезпеченні, так і в апаратних системах. Саме це сприяє широкому використанню XOR у криптографічних протоколах, гарантуючи швидкість та надійність процесу шифрування. Важливо відзначити, що ефективне застосування XOR безпосередньо залежить від якості ключових параметрів. Якщо ключ, який використовується для подальшого комбінування бітів, має високу ентропію, операція XOR забезпечує максимальний рівень захисту даних при малих змінах відкритого тексту. Таким чином, симетрична криптографія, яка інтегрує операцію XOR, дозволяє досягти високого ступеня дифузії та заплутування інформації, незамінного для стійкості криптосистем.

1.2 Визначення структури і властивостей криптоалгоритму DES

Алгоритм DES (Data Encryption Standard) історично став одним із перших широко застосовуваних симетричних шифрів завдяки своїй відносній простоті, високій швидкості обробки даних та досить високій криптографічній стійкості. Основні переваги DES полягають в тому, що:

- використовується лише один ключ ефективної довжини 56 біт (64 біти ключа, з яких 8 – біти контролю парності),
- для розшифрування зашифрованого блоку можна використати зворотний порядок операцій, незалежно від послідовності пакетів,

- відносна простота реалізації алгоритму забезпечує швидку обробку інформації,
- алгоритм демонструє достатній рівень стійкості до перебору ключа та інших методів криптоаналізу.

DES здійснює шифрування 64-бітових блоків даних за допомогою 56-бітового ключа. Розшифрування виконується як зворотна процедура шифрування, тобто операції шифрування повторюються в зворотному порядку. Стандартна послідовність дій алгоритму DES включає початкову перестановку бітів, 16 циклічних ітерацій (раундів) шифрування та фінальну зворотну перестановку. Ця схема, зображена на рис. 1.9, є типовою для всіх реалізацій DES, а всі таблиці, які наведені нижче, є стандартними і включаються без змін для забезпечення максимальної стійкості до підбору ключа.

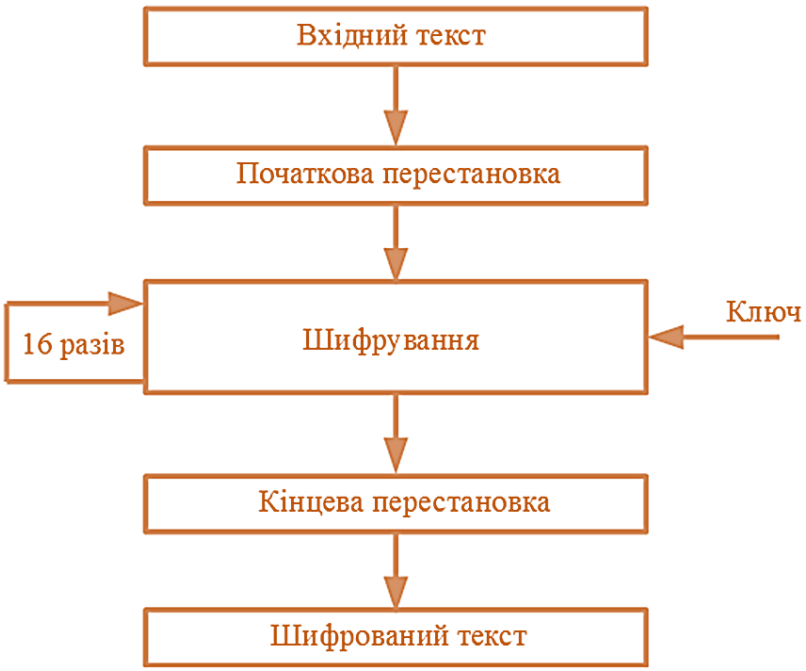


Рисунок 1.9. Стандартна послідовність дій алгоритму DES

1.2.1 Початкова обробка даних та структура шифрування DES

Нехай із зовнішнього файлу зчитується послідовність з 8 байтів (64 біт) — блок T. Далі цей блок підлягає початковій перестановці, реалізованій за допомогою матриці IP (табл. 1.4). За правилами табл. 1.4, наприклад, біт 58 блоку T переміщується на позицію 1, біт 50 – на позицію 2 і так далі, що дає в результаті перетворений блок T(0) = IP(T).

Отриману 64-бітову послідовність T(0) розбивають на дві частини по 32 біта:

- L(0) — ліва частина (старші біти),
- R(0) — права частина (молодші біти).

Таблиця 1.4. Матриця початкової перестановки IP у алгоритмі DES

58	50	42	34	26	18	10	02
60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06
64	56	48	40	32	24	16	08
57	49	41	33	25	17	9	01
59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05
63	55	47	39	31	23	15	07

Далі дані обробляються раундами шифрування. На кожній ітерації використовується функція шифрування f, що приймає два аргументи:

- 32-бітову послідовність R(i–1), отриману на попередньому кроці,
- 48-бітовий підключ K(i), який формується з первинного 64-бітового ключа K (з якого видаляються 8 біт контролю парності, що залишає 56 біт).

Після 16-ти раундів отримують послідовності L(16) та R(16), які конкатенують у 64-бітову комбінацію R(16) L(16) (зверніть увагу, що перестановка не виконувалася під час останнього раунду). Потім ця послідовність піддається зворотній перестановці, що реалізована за допомогою матриці IP^{-1} (табл. 1.5).

Таблиця 1.5. Матриця зворотної перестановки IP^{-1} у алгоритмі DES

40	08	48	16	56	24	64	32
39	07	47	15	55	23	63	31
38	06	46	14	54	22	62	30
37	05	45	13	53	21	61	29
36	04	44	12	52	20	60	28
35	03	43	11	51	19	59	27
34	02	42	10	50	18	58	26
33	01	41	9	49	17	57	25

Функція зворотної перестановки відновлює порядок бітів так, що, наприклад, перший біт матриці IP^{-1} дорівнює 40, а 40-й біт матриці IP дорівнює 1.

Процес розшифрування здійснюється як інверсія шифрування: дані, що підлягають розшифруванню, спочатку переставляються відповідно до IP^{-1}, а потім виконуються ті ж процедури шифрування, лише в зворотному порядку ($R(i–1) = L(i)$ для $i = 1,\ldots,16$). Після завершення ітерацій отримані послідовності L(0) та R(0) з'єднують у 64-бітовий блок, і застосовують перестановку IP для відновлення початкового тексту.

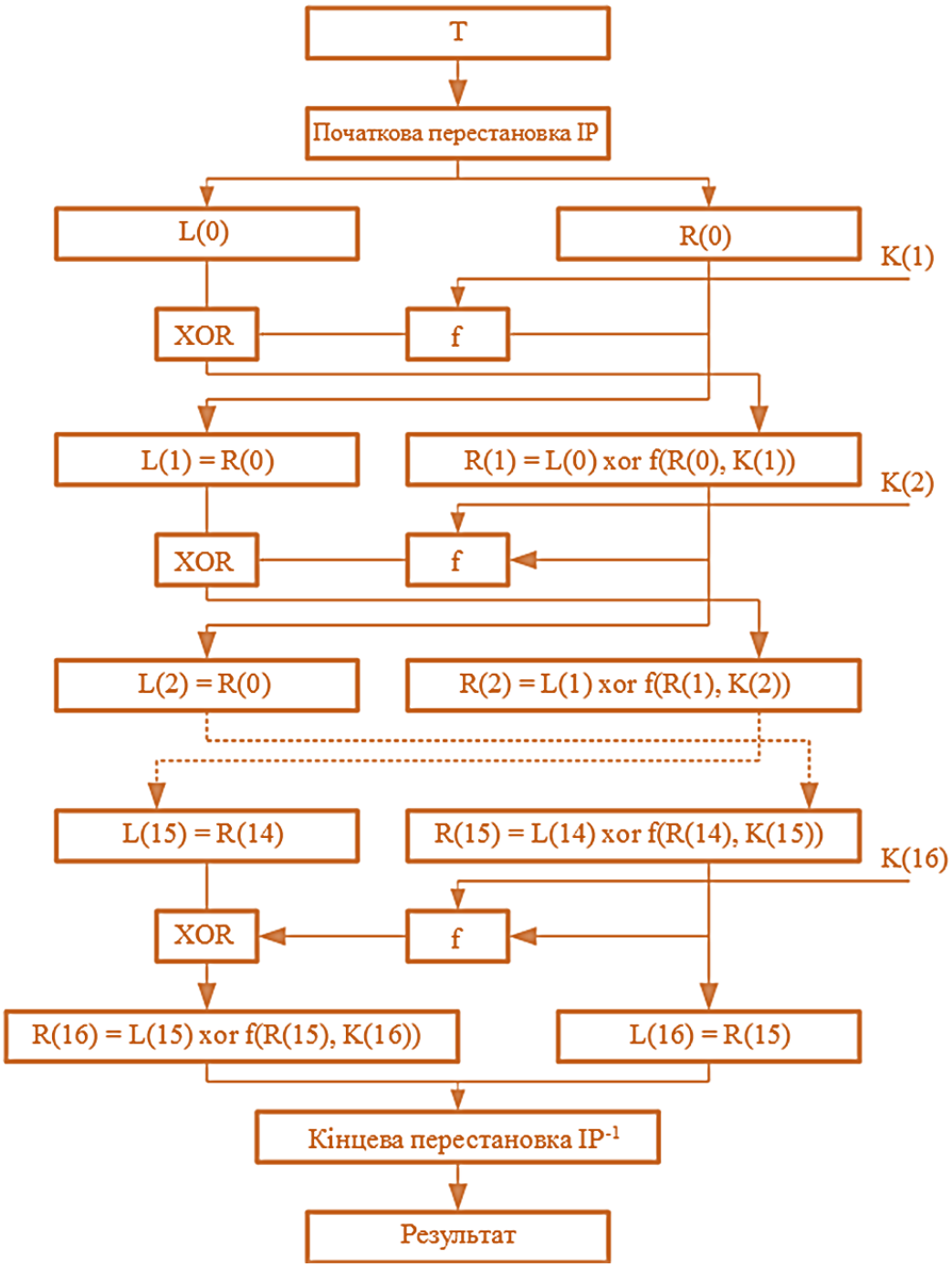


Рисунок 1.10. Схема шифрування у алгоритмі DES

Структура алгоритму DES узагальнено представлена на рис. 1.10, де зображено послідовність початкової перестановки, 16 раундів шифрування та фінальну зворотну перестановку, розбиття блоку та ітеративну обробку даних.

1.2.2 Функція шифрування f та її складові

Функція f(R(i–1), K(i)) є основною операцією, що забезпечує криптографічне перетворення даних. Вона виконується за наступним порядком:

1. Розширення. 32-бітова послідовність R(i–1) розширюється до 48 біт за допомогою функції E (табл. 1.6). За правилами табл. 1.6, перші три біти отримано з бітів 32, 1 і 2, а останні — з бітів 31, 32 і 1;

Таблиця 1.6. Функція розширення E у алгоритмі DES

32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	01

2. Комбінування з підключем. Розширена 48-бітова послідовність за допомогою операції XOR об'єднується з 48-бітовим підключем K(i). Результат цього поєднання розбивається на вісім 6-бітових блоків: E(R(i–1)) XOR K(i) = B(1) B(2) … B(8);
3. Підстановки (S-box). Кожен блок B(j) (де j = 1,…,8) обробляється відповідною функцією підстановки Sj (табл. 1.7). Якщо B(j) представлено як послідовність бітів $b_1b_2b_3b_4b_5b_6$, то двобітове число, утворене b_1 та b_6, визначає номер рядка, а число $b_2b_3b_4b_5$ — номер стовпця в матриці. Результатом є 4-бітове число, що записується у двійковій формі. Наприклад, якщо B(1) = 011011, то по правилах таблиці S1 значення, що знаходиться на перетині відповідного рядка та стовпця, може бути 0101;
4. Перестановка. Отримана 32-бітова послідовність, яка формується шляхом

конкатенації результатів S-box обчислень, піддається перестановці бітів за допомогою функції P (табл. 1.8). Функція P перемішує біти так, що, наприклад, біт 16 стає бітом 1, біт 7 – бітом 2 тощо.

Таблиця 1.7. Функції підстановки Sj у алгоритмі DES

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	S1
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	S2
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	S3
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	S4
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	S5
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	S6
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	S7
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	S8
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	

Таблиця 1.8. Функція перестановки P у алгоритмі DES

16	07	20	21
29	12	28	17
01	15	23	26
05	18	31	10
02	08	24	14
32	27	03	09
19	13	30	06
22	11	04	25

Таким чином, функцію шифрування можна записати як: $f(R(i–1), K(i)) = P(S_1(B(1)) \| S_2(B(2)) \| \ldots \| S_8(B(8)))$.

На рис. 1.11 схематично показано етапи визначення функції f: розширення, комбінування з підключем, підстановки та перестановку.

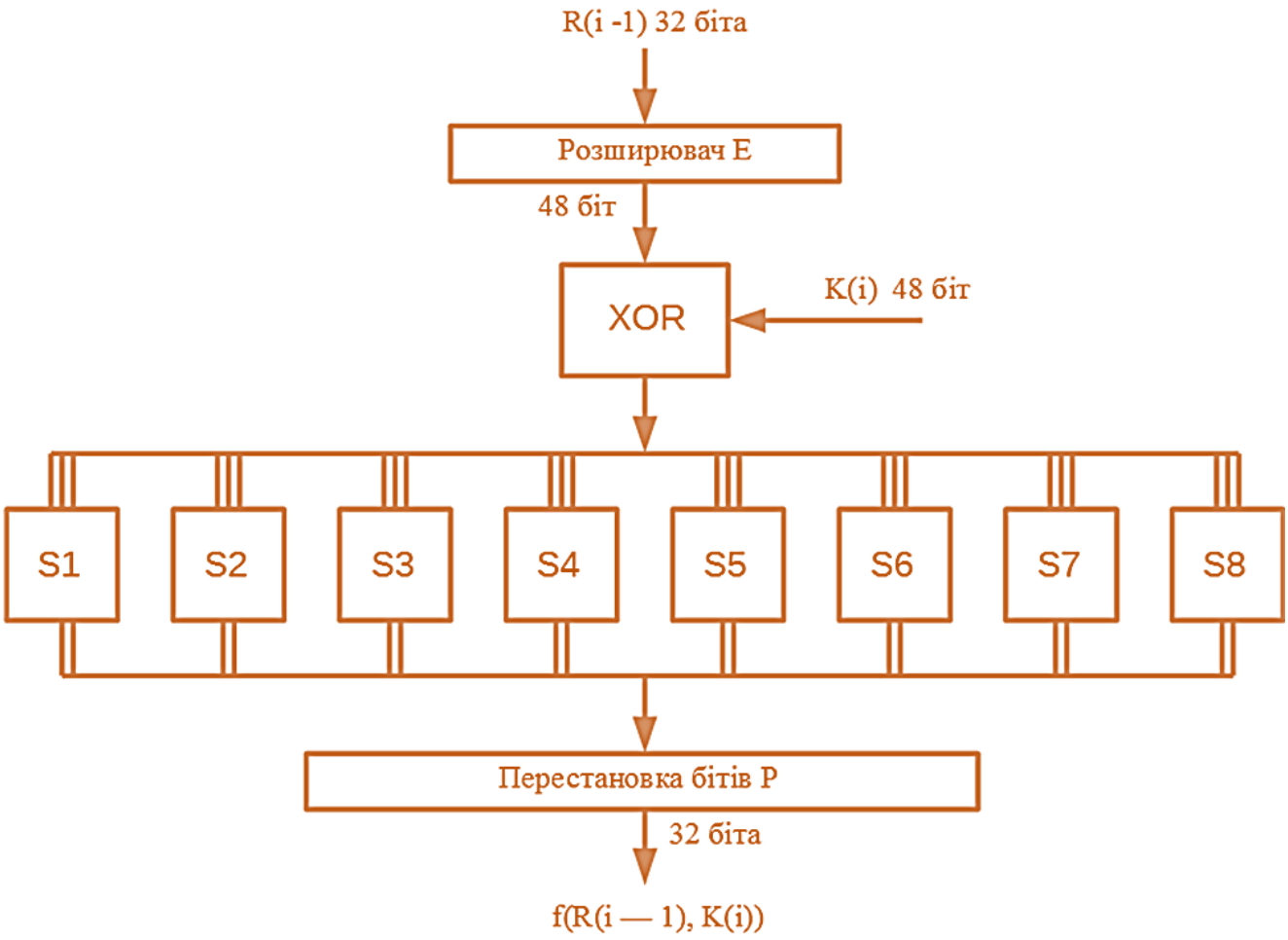


Рисунок 1.11. Обчислення функції шифрування f у алгоритмі DES

1.2.3 Генерація підключів: алгоритм розкладання ключа

Для DES вихідний 64-бітовий ключ містить 8 біт контролю парності (розташованих, наприклад, на позиціях 8, 16, 24, 32, 40, 48, 56 та 64), тому ефективна довжина ключа складає 56 біт. Генерація 48-бітових підключів K(i) для 16-ти раундів полягає у наступних кроках:

1. Первинна обробка ключа. Застосовується функція G (табл. 1.9), яка видаляє біти контролю парності й переставляє решту бітів.

Таблиця 1.9. Матриця G первинної підготовки ключа

57	49	41	33	25	17	09
01	58	50	42	34	26	18
10	02	59	51	43	35	27
19	11	03	60	52	44	36
63	55	47	39	31	23	15
07	62	54	46	38	30	22
14	06	61	53	45	37	29
21	13	05	28	20	12	04

Результат перетворення G(K) розбивається на два 28-бітових блоки:

- C(0) – блок, що містить біти з позицій 57, 49, …, 44, 36,
- D(0) – блок, що містить біти з позицій 63, 55, …, 12, 4.

2. Циклічні зсуви та рекурсивне обчислення. Для кожного з 16-ти раундів обчислюються нові версії блоків C(i) та D(i) за допомогою циклічних зсувів вліво. Зсув виконується на 1 або 2 біта залежно від номера ітерації, згідно з табл. 1.10.

Таблиця 1.10. Таблиця зсувів для обчислення ключа DES

Номер раунду:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Зсув (біт):	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

3. Фінальна обробка підключів. Після обчислення C(i) та D(i) їх конкатенують у 56-бітову послідовність, яку потім обробляють за допомогою матриці H (табл. 1.11) для отримання фінального 48-бітового підключа K(i).

Таблиця 1.11. Матриця H кінцевої обробки ключа

14	17	11	24	01	05
03	28	15	06	21	10
23	19	12	04	26	08
16	07	27	20	13	02
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

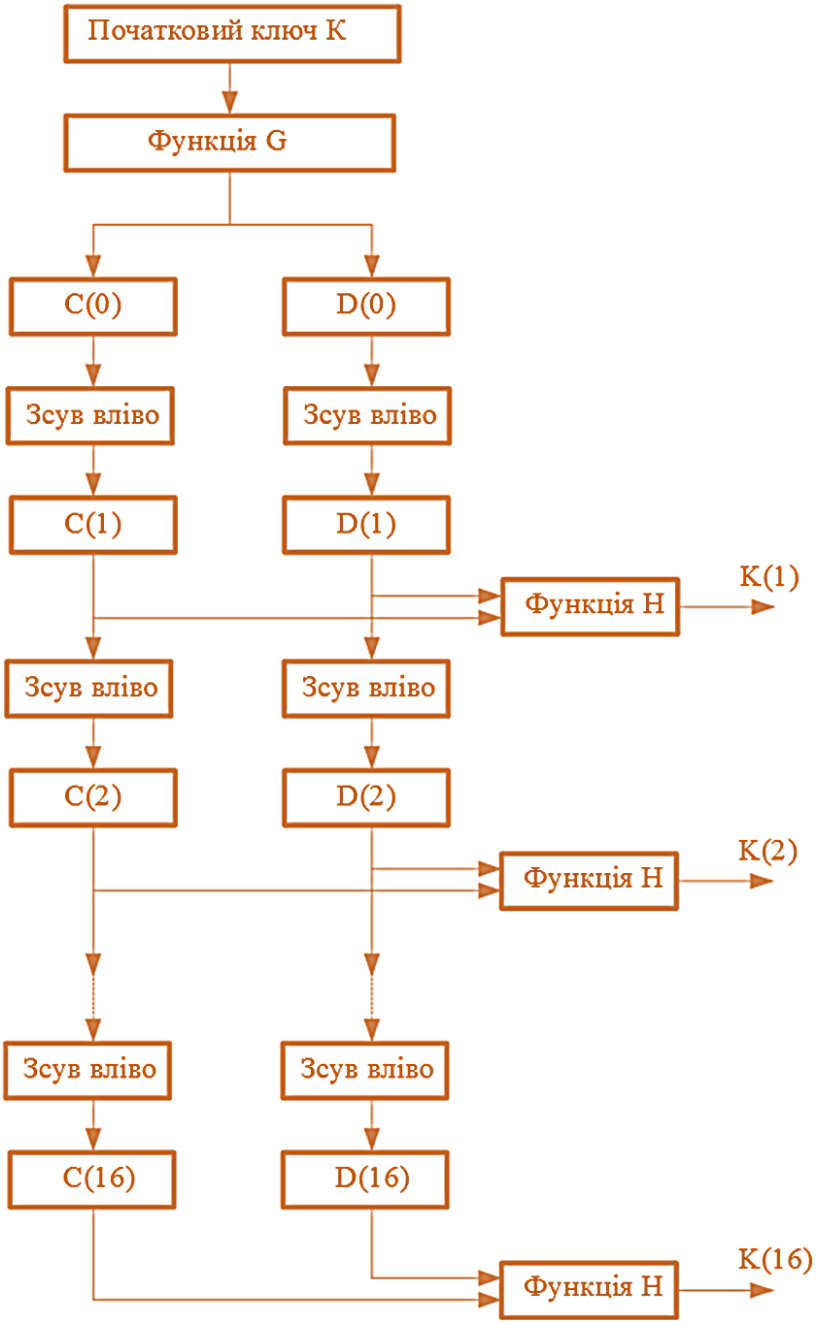


Рисунок 1.12. Генерування підключів у криптоалгоритмі DES

Підключ K(i) формується з бітів отриманої послідовності згідно з матрицею H. Блок-схема алгоритму генерації підключів представлена на рис. 1.12. Тут показано послідовність перетворень від вихідного ключа K до отримання 16 підключів K(1)…K(16).) Процес розшифрування здійснюється за тим же алгоритмом, проте підключі застосовуються у зворотному порядку (спочатку K(16), потім K(15) і т.д.).

Алгоритм DES, таким чином, характеризується наступними властивостями:

- Шифрування 64-бітових блоків здійснюється за допомогою 56-бітового ключа, що забезпечує компактність і ефективність;
- Операції шифрування базуються на початковій перестановці, 16 послідовних раундах, де використовується функція шифрування f (з розширенням, підстановками та перестановкою), та фінальній зворотній перестановці;
- Генерація підключів здійснюється за допомогою попередньої обробки, циклічних зсувів та остаточного «перемішування» за матрицею H, що ускладнює процес відновлення початкового ключа шляхом перебору;
- Стандартні таблиці (IP, IP^{-1}, E, S-box, P, G, H, а також таблиця зсувів) використовуються без змін, що гарантує відповідність алгоритму оригінальному стандарту.

1.3 Аналіз існуючих модифікацій криптоалгоритму DES

У цьому підрозділі розглядаються основні модифікації DES, які зберігають базову структуру Фейстелевої мережі, але змінюють окремі елементи з метою підвищення стійкості до криптоаналізу та розширення ключового простору.

1.3.1 Криптоалгоритм Triple DES (3DES)

Однією з найбільш визнаних модифікацій є Triple DES (3DES), або TDEA (Triple Data Encryption Algorithm). Основна ідея 3DES полягає у застосуванні алгоритму DES тричі послідовно для кожного 64-бітового блоку даних—зазвичай у режимі "Encrypt-Decrypt-Encrypt" (EDE). Завдяки цьому ефективна довжина ключа збільшується до 112 або 168 біт (в залежності від використання 2 або 3 ключів відповідно), що значно підвищує стійкість алгоритму до перебору ключа.

Відповідно, хоча базова структура DES (див. рис. 1.10) залишається незмінною, повторне застосування операцій шифрування у 3DES забезпечує додатковий рівень захисту (рис.1.13, 1.14).

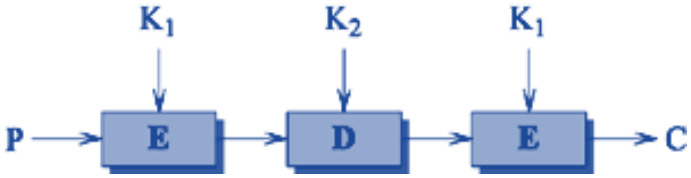


Рисунок 1.13. Принцип шифрування криптоалгоритмом 3DES

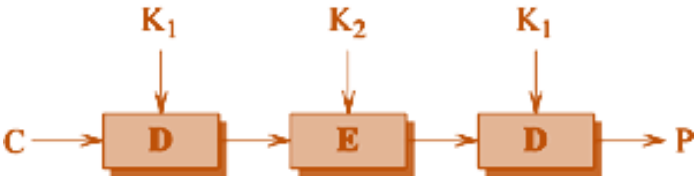


Рисунок 1.14. Принцип дешифрування криптоалгоритмом 3DES

1.3.2 Криптоалгоритм DES-X та інші методи "whitening"

Іншою популярною модифікацією є DES-X, яка інтегрує процес, відомий як "key whitening". У цьому випадку перед початком основного циклу шифрування і після нього здійснюється операція XOR з додатковими ключовими блоками. Цей додатковий рівень змішування не змінює внутрішню структуру механізму DES, але суттєво збільшує ефективну довжину ключа, що ускладнює атаки методом перебору. Такий підхід дозволяє комбінувати базову процедуру DES (див. рис. 1.10) з додатковим шифруванням, що підвищує стійкість алгоритму без значного збільшення обчислювальних витрат.

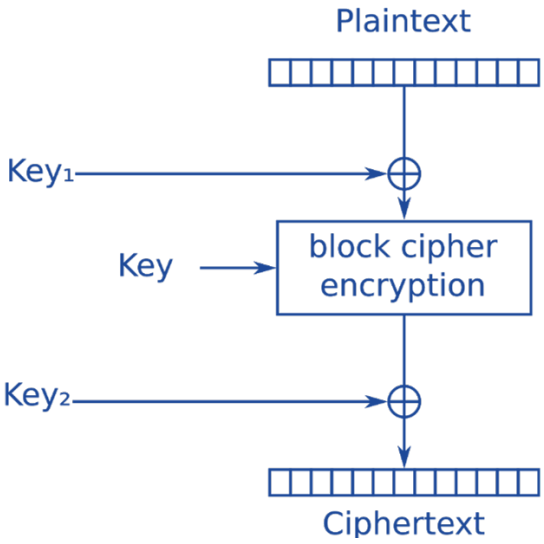


Рисунок 1.14. Принцип шифрування криптоалгоритмом DES-X

1.3.3 Модифікації структури S-блоків та розширення ключового простору

У деяких модифікаціях DES розробники пропонують змінити або частково перепроектувати таблиці заміщення (S-блоки), які відіграють важливу роль у забезпеченні нелінійності функції шифрування f (див. рис. 1.11 та табл. 1.7). Такі зміни спрямовані на зниження ефективності диференціального криптоаналізу, який здатен експлуатувати властивості умовного переходу в традиційних S-блоках DES. Хоча оригінальні S-блоки були ретельно підібрані розробниками алгоритму, деякі дослідники запропонували варіанти з більш гнучкою структурою, що дозволяють адаптувати алгоритм до сучасних стандартів безпеки. Крім того, існують модифікації, що розширюють ключовий простір за рахунок використання додаткових перестановок або зміни алгоритму генерації підключів (див. табл. 1.9–1.11, рис.1.12), що призводить до ефективної довжини ключа, більш стійкої до атак перебором.

1.3.4 Альтернативні варіанти кількості раундів та побудови ключового графа

Ще одним напрямком модифікацій DES є варіації у кількості раундів шифрування. Хоча стандартний DES працює з 16-ма раундами, деякі дослідники експериментували з додатковими раундами для підвищення стійкості алгоритму до атаки методом диференціального криптоаналізу. Таке розширення зберігає базову ідею Фейстелевої мережі (див. рис. 1.6), але підвищує складність повторного відновлення ключа. Також пропонуються варіанти з модифікованими процедурами циклічного зсуву бітів та розкладання ключа, що дозволяють створити змінну ключову матрицю, не знищуючи оборотності конструкції, але роблячи її більш адаптивною до сучасних вимог.

1.3.5 Порівняння модифікованих версій DES з алгоритмом AES

Криптостійкість алгоритмів шифрування визначається низкою факторів: довжиною ключа, стійкістю до криптоаналізу, продуктивністю, структурними особливостями та можливістю ефективної програмної чи апаратної реалізації. У цьому підрозділі проведено порівняння модифікованих версій алгоритму DES

(зокрема 3DES, DES-X) з сучасним алгоритмом AES (Advanced Encryption Standard) за ключовими характеристиками.

Однією з основних проблем оригінального DES була недостатня довжина ключа — 56 біт, що робило його вразливим до атаки методом повного перебору. Модифіковані версії DES, такі як 3DES (Triple DES), значно розширюють ключовий простір, використовуючи три послідовні шифрування з довжиною ключа 112 або 168 біт.

Алгоритм AES, у свою чергу, підтримує ключі довжиною 128, 192 або 256 біт, що робить його більш захищеним від атак типу brute-force. Крім того, AES демонструє значно вищу швидкість при більших об'ємах даних, оскільки його структура була оптимізована під сучасні процесори.

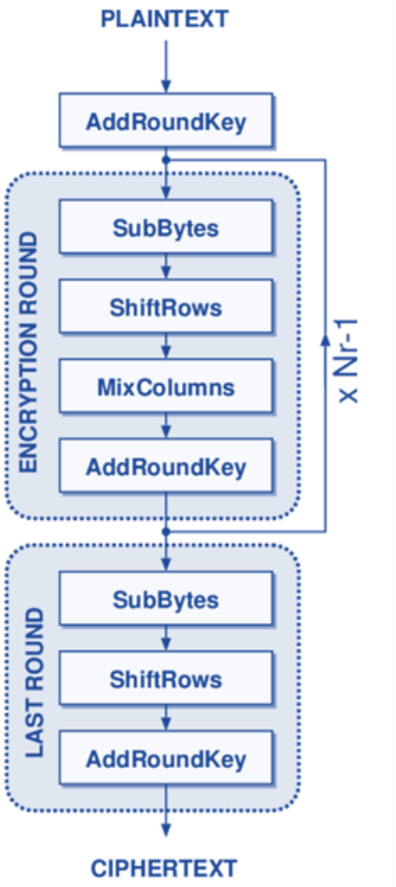


Рисунок 1.15. Реалізація шифрування за криптоалгоритмом AES

Одна з ключових відмінностей AES від DES — його конструкція. DES та його модифікації базуються на Фейстелевій мережі, що використовує перестановки та нелінійні S-блоки для обробки даних. У той же час AES базується на принципах заміни та перестановки без використання класичної Фейстелевої мережі.

Основними функціями AES є:

- SubBytes – таблична заміна кожного байта за наперед визначеною таблицею (на відміну від DES, де таблиці можуть змінюватися);
- ShiftRows – циклічний зсув рядків блоку;
- MixColumns – матричне перетворення на основі поля Галуа $GF(2^8)$, що забезпечує дифузію;
- AddRoundKey – побітова операція XOR з відповідним байтом ключа для кожної ітерації.

Як видно з рис. 1.15, структуру AES можна розглядати як матричну модель, у якій всі математичні операції виконуються над блоками даних у вигляді 4×4 матриць. На відміну від 3DES, який здійснює три цикли DES, AES використовує іншу підхідність обчислень, що дозволяє досягти швидшої обробки.

Існує кілька основних видів криптоаналізу, спрямованих на знаходження слабких місць в алгоритмах шифрування:

- Диференціальний криптоаналіз (аналіз залежностей між відкритим текстом і шифротекстом);
- Лінійний криптоаналіз (відновлення частини ключа на основі статистичних закономірностей);
- Атаки на ключовий потік (спрямовані на дешифрування повторюваних шифрованих блоків).

Оригінальний DES має вразливість до диференціального криптоаналізу через передбачуваність його S-блоків. Удосконалений DES-X вирішує цю проблему завдяки додатковій операції XOR з "білим ключем" (whitening). AES значно краще захищений від подібних атак через його нелінійні перетворення у функціях SubBytes та MixColumns, що унеможливлює прості математичні моделі відновлення ключа. Крім того, на відміну від DES, AES не демонструє вразливість до атак типу meet-in-the-middle, які були ефективними для 3DES.

Швидкість шифрування в AES значно вища порівняно з 3DES, що пояснюється більш ефективною структурою обробки блоків та оптимізацією для сучасних процесорів. Основна причина цього полягає в тому, що 3DES проводить

три цикли DES, що істотно збільшує витрати на обчислення. З іншого боку, AES використовує байтові операції, які легко реалізовуються на апаратному рівні, що дозволяє йому бути значно ефективнішим у високошвидкісних мережах та операціях із великими масивами даних. Недоліком AES є його вразливість до атак через сторонні канали (наприклад, аналіз витоку даних у пам'яті).

Узагальнене порівняння модифікованих версій DES із AES представлено в табл. 1.12:

Таблиця 1.12. Порівняння DES (3DES, DES-X) та AES

Характеристика	*DES / 3DES / DES-X*	*AES*
Ключовий простір	56 / 112 / 168 біт	128 / 192 / 256 біт
Структура	Фейстелева мережа	Таблиці, матричні перетворення
Кількість раундів	16 / 48	10 / 12 / 14
Операції шифрування	S-блоки, XOR, перестановки	SubBytes, ShiftRows, MixColumns, XOR
Швидкість	Низька для 3DES	Висока
Реалізація	Оптимізована для ПЗ	Швидка на апаратному рівні

1.3.6 Висновки щодо модифікацій криптоалгоритму DES

Сучасні модифікації криптоалгоритму DES спрямовані переважно на подолання недоліків оригінального стандарту, зокрема недостатньої довжини ключа та вразливості до перебору. Основні напрямки модернізації включають:

- Збільшення ефективної довжини ключа: реалізація 3DES та DES-X дозволяє значно розширити ключовий простір без масштабної зміни базової структури алгоритму;
- Покращення нелінійності: зміни в структурі S-блоків та застосування додаткових перестановок сприяють посиленню заплутування даних;
- Оптимізація процедури генерації підключів: різноманітні варіанти key scheduling дозволяють адаптувати алгоритм до сучасних вимог без шкоди для оборотності шифрування.

Хоча алгоритм AES зараз є стандартом у більшості криптографічних застосувань, модифіковані версії DES, зокрема 3DES та DES-X, все ще знаходять

своє практичне застосування, зокрема:

- Системи, що вимагають зворотної сумісності. Велика кількість застарілих систем у фінансовій сфері, банківських терміналах, системах обробки платіжних карток та інших промислових застосуваннях досі використовує модифіковані версії DES. Це зумовлено необхідністю підтримувати інтеграцію з існуючим обладнанням, сертифікованим для використання стандартного DES чи його модифікацій;

- Вбудовані системи та апаратні засоби. У деяких апаратних рішеннях, зокрема в мікроконтролерах та пристроях з обмеженими ресурсами, модифіковані DES версії застосовують завдяки їх відносній простоті реалізації та низьким вимогам до обчислювальної потужності. Це дозволяє швидко виконувати шифрування без значного навантаження на систему;

- Освітні та дослідницькі проекти. Завдяки відкритості алгоритмічної структури DES модифікації цього алгоритму часто використовують для навчання принципам симетричного шифрування, аналізу криптографічних методів та моделювання криптоаналітичних атак. Це дозволяє отримати практичний досвід зі взаємодії між різними компонентами криптосистем;

- Перехідні рішення в критичних галузях. У випадках, коли заміна існуючих систем на сучасні стандарти (як AES) є складною або витратною, використання 3DES або DES-X дозволяє протягом певного періоду забезпечувати достатній рівень захисту, одночасно підготувавши інфраструктуру до повного переходу на новітні технології.

1.4 Аналіз технічного завдання і постановка задачі проектування

Дипломний проект передбачає розробку криптографічного кодеку на основі модифікованого алгоритму DES. Основними завданнями є:

- Вибір оптимального модифікованого алгоритму DES. При цьому головну увагу слід звернути на підвищення рівня безпеки за рахунок використання покращених варіантів класичного алгоритму. DES вважається застарілим через коротку довжину ключа, що робить його вразливим до сучасних атак.

- Моделювання кодеку у середовищі CrypTool2. Завдання полягає у створенні симуляції алгоритму для аналізу його поведінки, оцінки ефективності та стійкості до криптоаналітичних методів. CrypTool2 забезпечує достатній набір інструментів для детального моделювання, що дозволяє отримати наочне представлення роботи алгоритму.

- Програмна реалізація кодеку за допомогою Visual Studio мовою C#. Після отримання необхідних параметрів і випробувань у середовищі CrypTool2, реалізація алгоритму переходить у фазу практичної адаптації, де важливо забезпечити зрозумілий інтерфейс, модульну архітектуру та відповідність сучасним стандартам програмування.

Метою проектування є моделювання роботи кодеку для шифрування/розшифрування текстових файлів, заснованого на модифікованому варіанті DES. Для цього необхідно вирішити наступні завдання:

1. Вибір оптимального алгоритму: Аналіз існуючих модифікацій DES показує, що класичний алгоритм не відповідає сучасним вимогам безпеки. Саме тому у даному проекті обрано алгоритм 3DES (Triple DES) як оптимальну модифікацію. Використання трьох послідовних операцій шифрування (або послідовності шифрування-розшифрування-шифрування) дозволяє значно збільшити стійкість до криптоаналізу, оскільки ефективна довжина ключа зростає до 168 біт, що суттєво ускладнює атаки методом перебору та інші сучасні криптографічні методи зламу;

2. Обґрунтування вибору 3DES з сімейства алгоритмів DES:

- Безпека: Порівняно з базовим DES, алгоритм 3DES є набагато більш стійким завдяки трьом послідовним етапам шифрування із застосуванням різних ключів. Це дозволяє уникнути вразливостей, притаманних одноразовому DES;

- Перевіреність: 3DES використовується протягом багатьох років у промислових застосуваннях, що забезпечує його надійність та перевіреність у реальних умовах. Це важливо для академічного проекту, орієнтованого на демонстрацію технології;

- Сумісність з Cryptool 2.1: Середовище CrypTool2 має готові модулі для моделювання як традиційного DES, так і його модифікацій. Це дозволяє швидко

налаштувати симуляцію алгоритму 3DES та отримати змогу проводити порівняльний аналіз різних підходів;

- Простота інтеграції у Visual Studio: Реалізацію алгоритму на мові C# із використанням Visual Studio спрощує подальшу підтримку коду та інтеграцію у більші програмні системи завдяки високому рівню модульності та великій спільноті розробників, що використовують дане середовище;

3. Моделювання у CrypTool2: Планується створення детальної схеми роботи алгоритму, що включає:

- Побудову блок-схеми алгоритму із зазначенням основних операцій (шифрування, розшифрування, генерація ключів);

- Експериментальний аналіз ефективності алгоритму при різних конфігураціях ключів;

- Отримання статистичних показників, які дадуть змогу вибрати налаштування з оптимальним співвідношенням продуктивності та безпеки;

4. Програмна реалізація у Visual Studio мовою C#: Після моделювання алгоритму основну увагу буде приділено переведенню отриманих результатів у робочий код:

- Створення графічного застосунку для демонстрації роботи кодеку;

- Забезпечення модульності програмного коду, щоб кожен етап (генерація ключів, шифрування, розшифрування) міг бути протестований окремо;

- Ретельне документування коду та створення тестових випадків для перевірки коректності реалізації.

Завдання проекту полягає у реалізації алгоритму Triple DES як оптимальної модифікації DES, що забезпечує високий рівень криптографічної стійкості завдяки збільшенню ефективної довжини ключа та перевіреним механізмам шифрування. Моделювання в CrypTool2 дозволить отримати наочну картину роботи алгоритму, а наступна програмна реалізація у Visual Studio із застосуванням мови C# забезпечить створення практичного та функціонального кодеку. Результати даного етапу відкриють можливості для подальших оптимізацій, впровадження додаткових модифікацій і розширення функціоналу системи.

1.5 Моделювання роботи алгоритму Triple DES у CrypTool2

Для перевірки коректності алгоритму Triple DES перед його подальшою програмною реалізацією було проведено моделювання роботи шифрувальної схеми з використанням програмного середовища CrypTool2. Це дозволило не лише підтвердити ефективність застосування трьох послідовних перетворень (шифрування-дешифрування-шифрування), а й продемонструвати покрокову логіку процесу шифрування даних. На рис. 1.16 зображено результати симуляції алгоритму Triple DES для відкритого тексту: “Доброго дня, шановна комісія. Мене звуть Бодачевський Даніїл”. В результаті отримано шифротекст “4E DB 57 E2 EA 5B C7 0E 06 4C F0 5F 19 60 CE 73 BF B2 13 19 21 EA 67 F2 D7 3B 3C A0 0F 97 37 D9 B9 7B 21 21 C0 2E E1 27 D1 65 00 A9 33 DD 44 28 B0 5C 3E 91 4D 62 DA 90 9F C4 64 4C 95 AD D2 00 BC CA 7F 2A 65 93 88 9E DB 72 6B 18 23 3E 00 A2 86 37 05 4C 87 16 28 7A 0C AE FF D2 72 A5 96 83 83 36 6F B6 1D 70 36 D2 DD 58 4A 95 98 AB 62 49”.

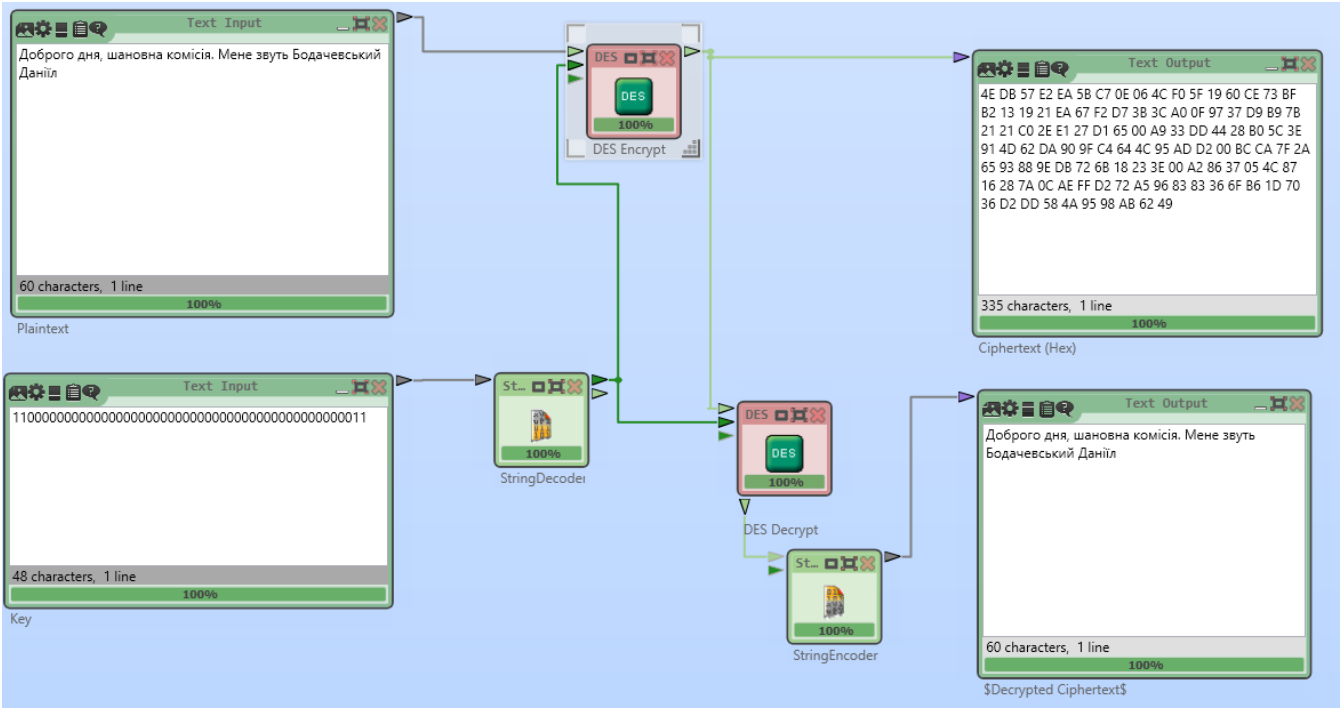


Рисунок 1.16. Результат моделювання у CrypTool2 за алгоритмом Triple DES

Цей результат свідчить про правильність реалізації алгоритму, адже послідовність трьох криптографічних операцій забезпечує високий рівень захисту за рахунок значного ускладнення DES.

На рис.1.17-1.24 показані окремі кроки моделювання за алгоритмом Triple DES при шифрування у покроковому режимі роботи програмного застосунку CrypTool2.

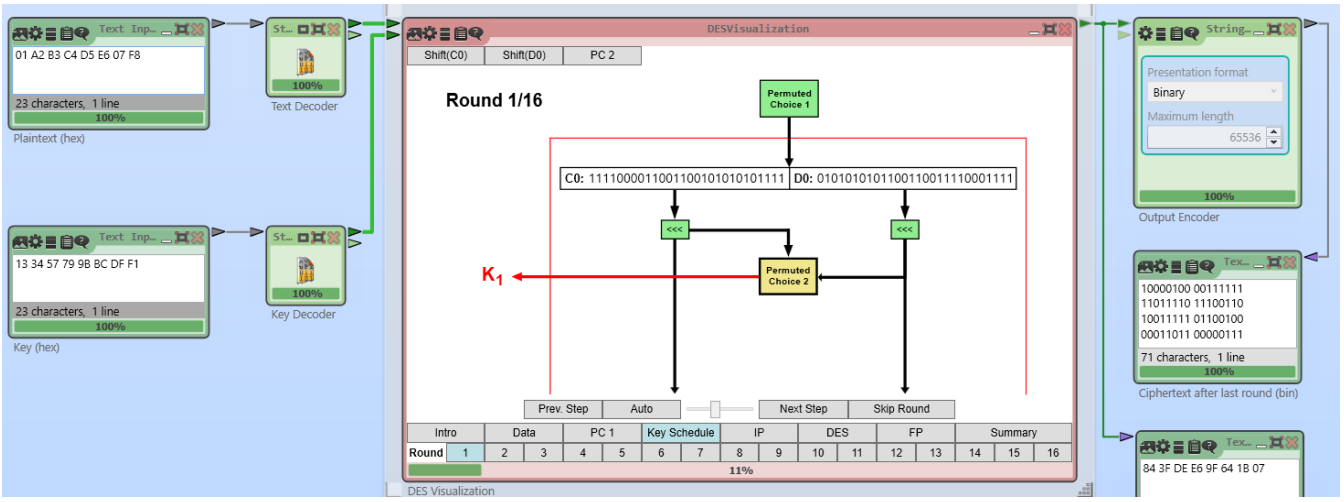


Рисунок 1.17. Перший раунд вибору ключу за алгоритмом DES

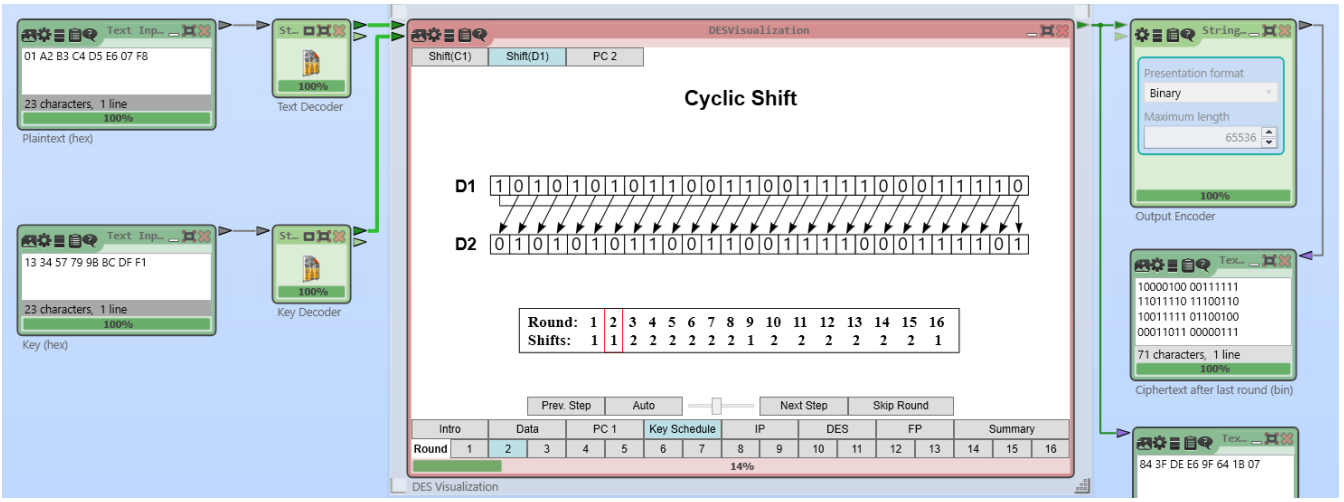


Рисунок 1.18. Циклічний зсув для ключу за алгоритмом DES

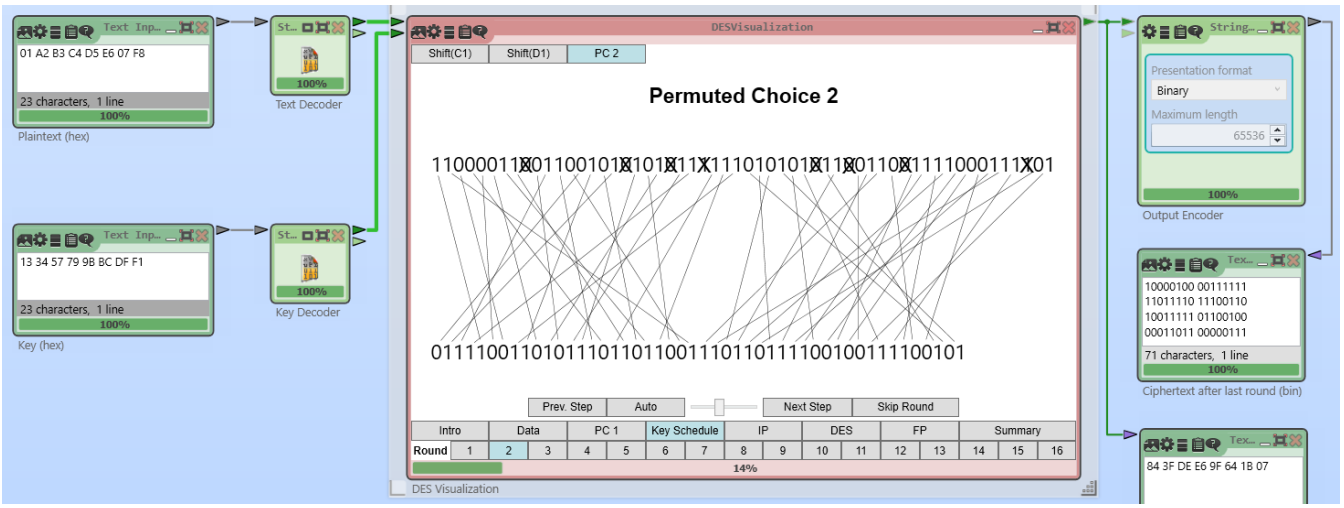


Рисунок 1.19. Перестановка для ключу за алгоритмом DES

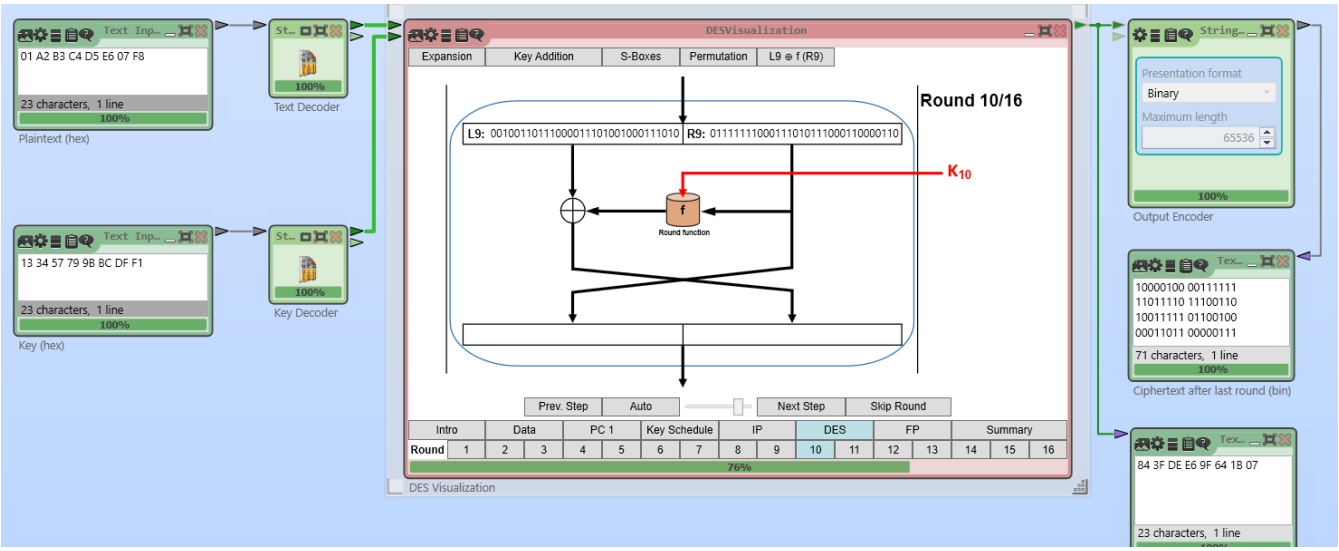


Рисунок 1.20. Раунд шифрування за алгоритмом DES

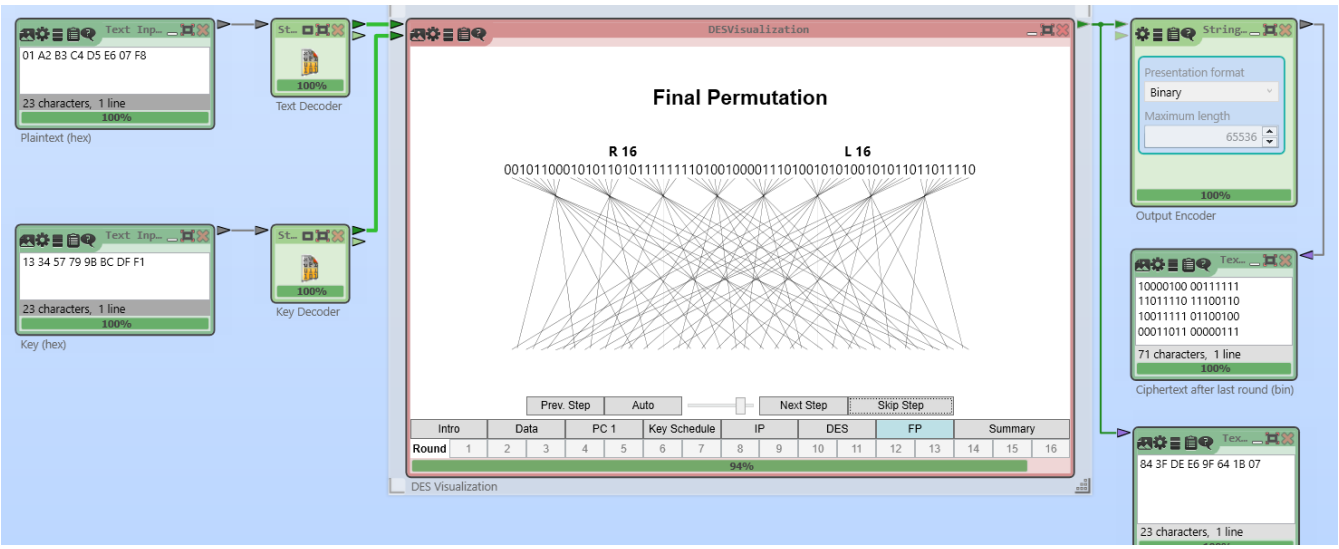


Рисунок 1.21. Кінцева перестановка за алгоритмом DES

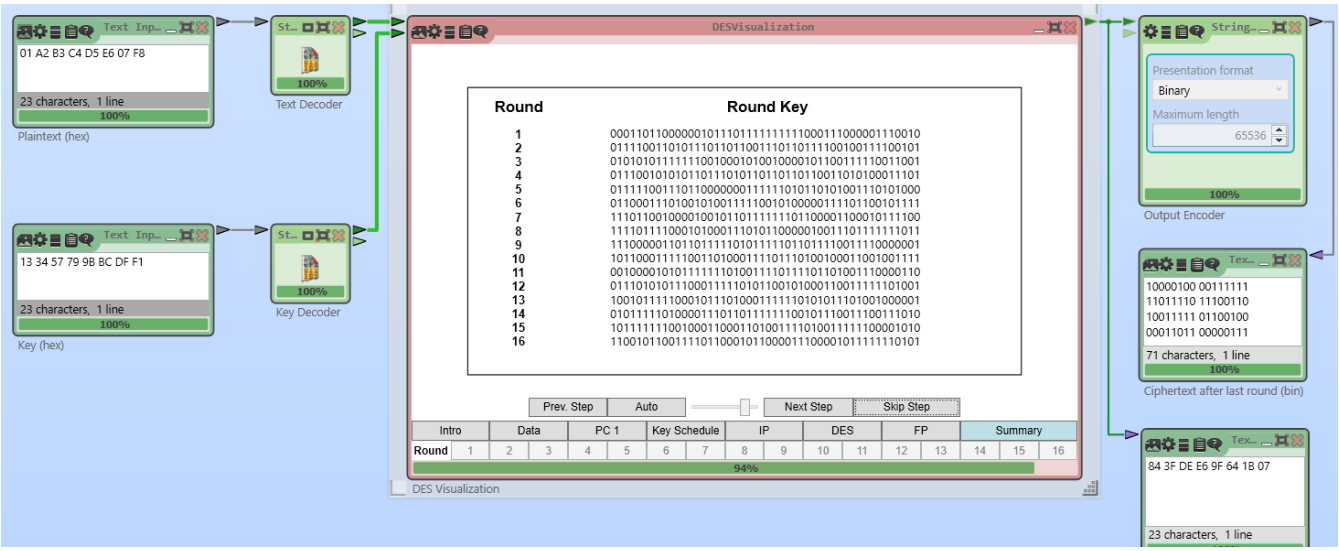


Рисунок 1.22. Раундові ключі для всіх 16 раундів алгоритму DES

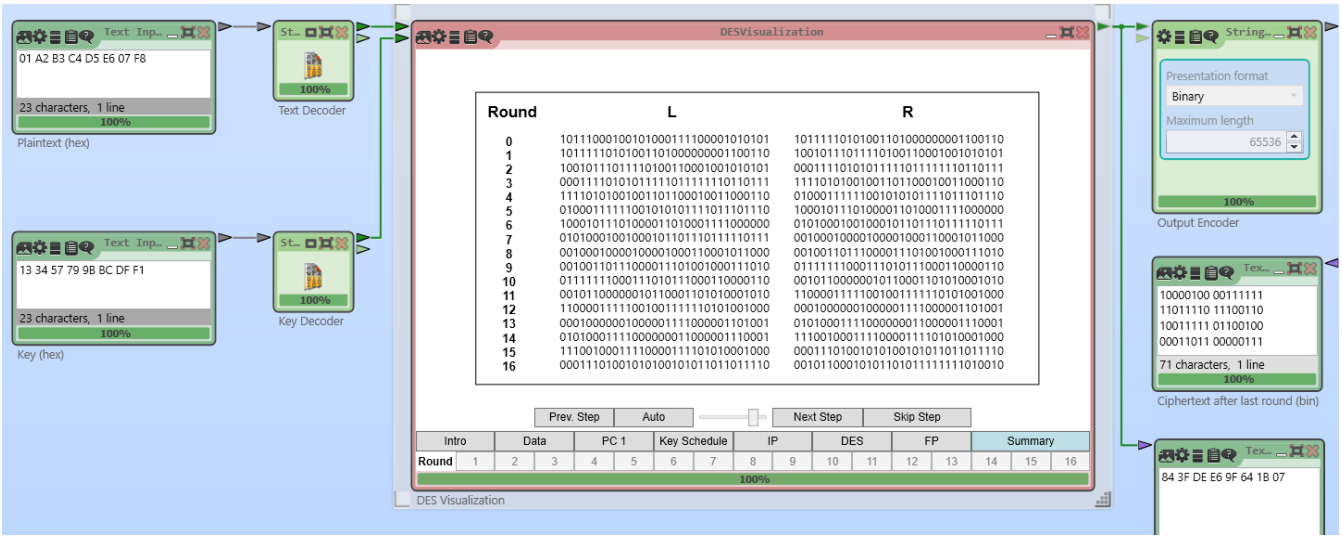


Рисунок 1.23. Ліві та праві частини блоку даних на всіх 16 раундах алгоритму DES

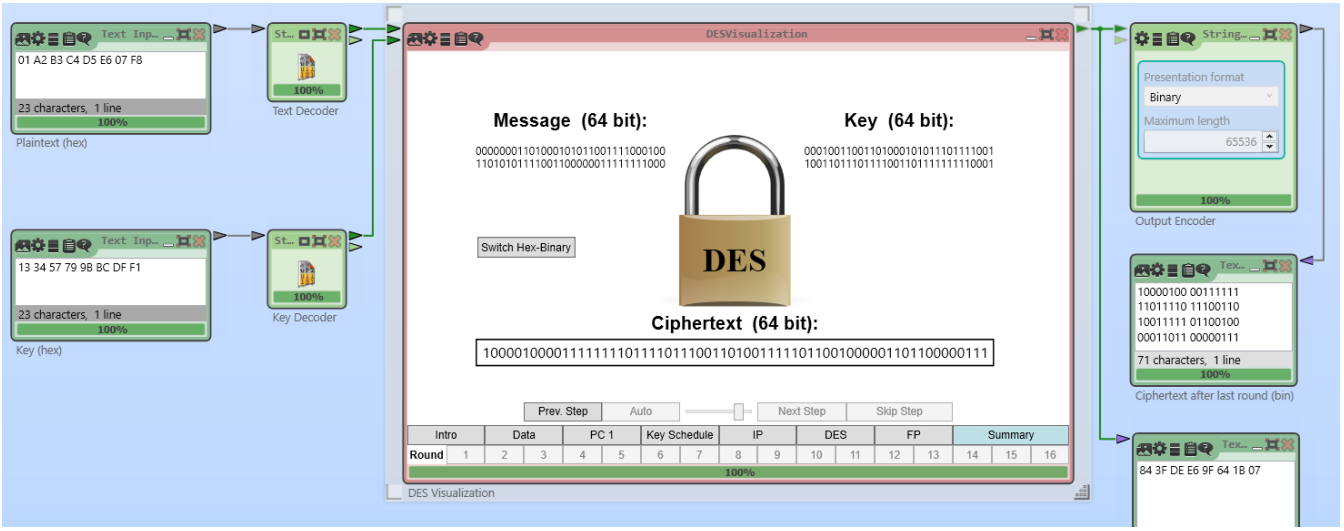


Рисунок 1.24. Отриманий блок шифротексту за алгоритмом DES

Моделювання у CrypTool2 дозволило детально прослідкувати за ходом виконання кожного етапу алгоритму Triple DES, перевірити логіку обчислень та верифікувати відповідність отриманого шифротексту заданим параметрам безпеки. Отримані результати слугують основою для подальшої розробки програмної реалізації кодеку у Visual Studio мовою C#.

1.6 Підготовка алгоритмів для програмної реалізації кодека Triple DES

В основі розробки програмної реалізації кодека Triple DES лежить детальний аналіз алгоритму DES та його подальше розширення у варіант із потрійним застосуванням, що дозволяє значно посилити криптографічний захист даних.

Класичний алгоритм DES працює з 64-бітовими блоками даних і використовує 8-байтний ключ, де один байт виділено для перевірки парності. Він базується на структурі мережі Фейстеля, за якої початково здійснюється перестановка вхідного блоку (рис. 1.25), після чого дані розбиваються на два 32-бітові підблоки та проходять через 16 раундів перетворень. Кожен раунд включає генерацію окремих 48-бітових підключів, розширення правої частини блоку, застосування операції XOR, обчислення через S-коробки для заміщення та виконання додаткової перестановки. Такий підхід забезпечує значну нелінійність та складність для криптоаналізу.

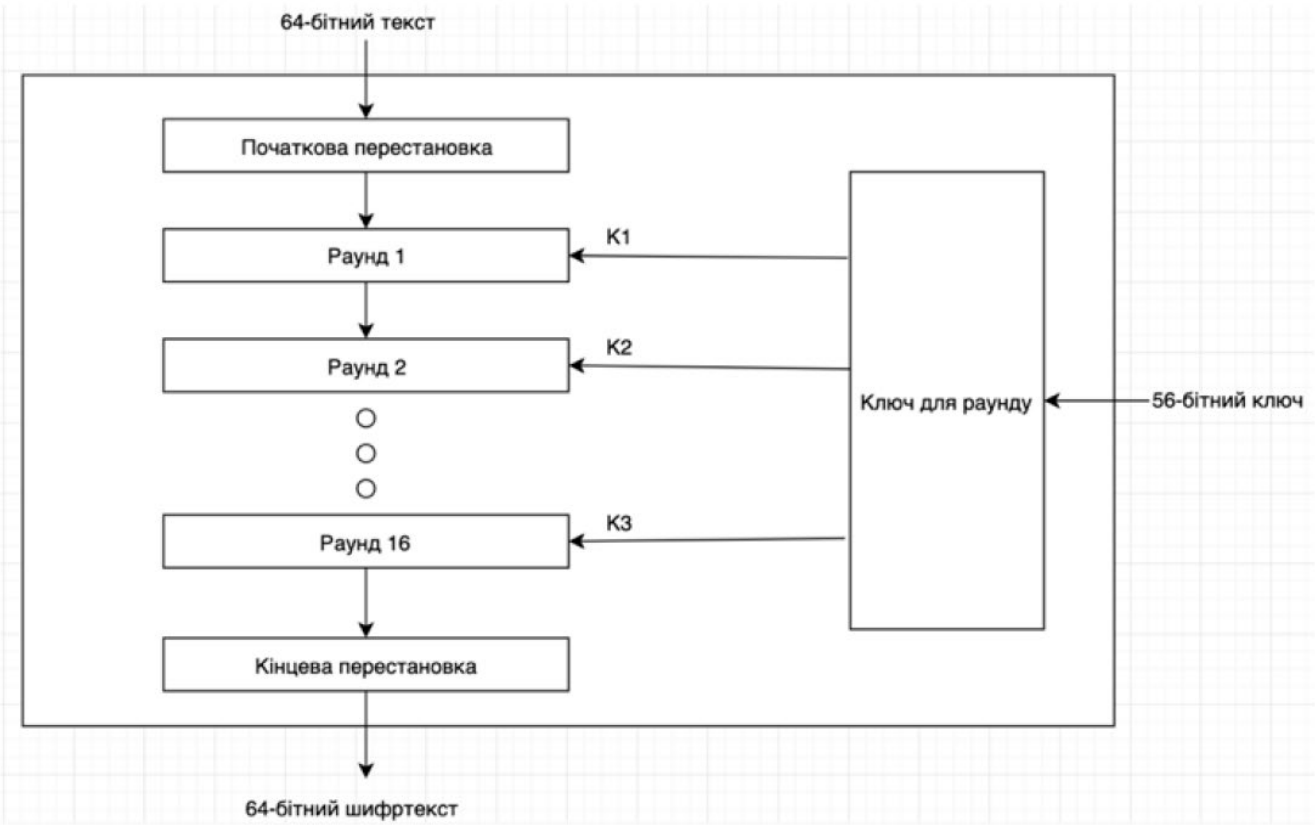


Рисунок 1.25. Структурна схема роботи криптоалгоритму DES

Перехід до алгоритму Triple DES передбачає послідовне виконання операцій шифрування та дешифрування за базовими принципами DES, що дозволяє реалізувати схему «шифрування – дешифрування – шифрування» (EDE). Використання трьох незалежних ключів, що в сумі формують 168-бітовий параметр, дозволяє усунути вразливості одиночного DES та гарантувати високий рівень захисту. При цьому можливі різні варіанти ключових пакетів, де оптимальним в плані безпеки є варіант із повністю незалежними ключами K1, K2 і K3. Така схема

не лише повторює логіку DES, а й інтегрує додаткові перетворення, що допомагають подвоїти складність розшифрування для зловмисника (рис.1.26).

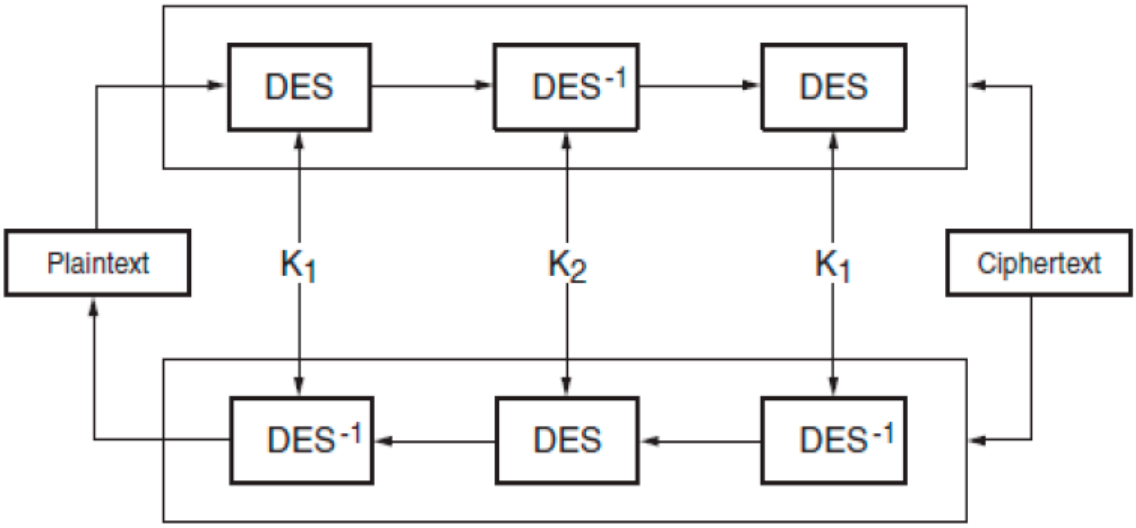


Рисунок 1.26. Структурна схема роботи криптоалгоритму Triple DES

Для подальшої програмної реалізації кодека Triple DES у середовищі Visual Studio мовою C# розробку алгоритмів планується організувати у вигляді модульних компонентів, що дозволить окремо реалізувати, протестувати та інтегрувати кожну частину процесу. Зокрема, реалізація передбачає розробку класів для генерації ключів із перевіркою парності, реалізацію функціональних модулів для початкової перестановки вхідного блоку та розбиття його на дві 32-бітові частини, а також написання єдиної функції, що інкапсулює логіку одного раунду алгоритму DES. Ця функція реалізує розширення правої частини до 48 бітів, виконує операцію XOR з відповідним підключем, підставляє результати через S-коробки для отримання 32-бітного результату, а також проводить фінальну перестановку. Надалі, реалізація моделі Triple DES включатиме інтеграцію даної функції у послідовність операцій, де спочатку дані шифруються за допомогою першого ключа K1, потім дешифруються із застосуванням ключа K2, і, нарешті, ще раз шифруються із застосуванням ключа K3.

Для візуалізації та кращого розуміння логіки програми також було розроблено UML-діаграми, що відображають процес діяльності користувача під час роботи системи. Зокрема, UML-діаграма діяльності (рис. 1.27) демонструє послідовність етапів, що починається з підготовки до обрахунку, вибору типу вхідних даних,

завантаження файлу з алгоритмом та введення додаткових параметрів, і завершується підтвердженням даних перед початком фактичного процесу шифрування.

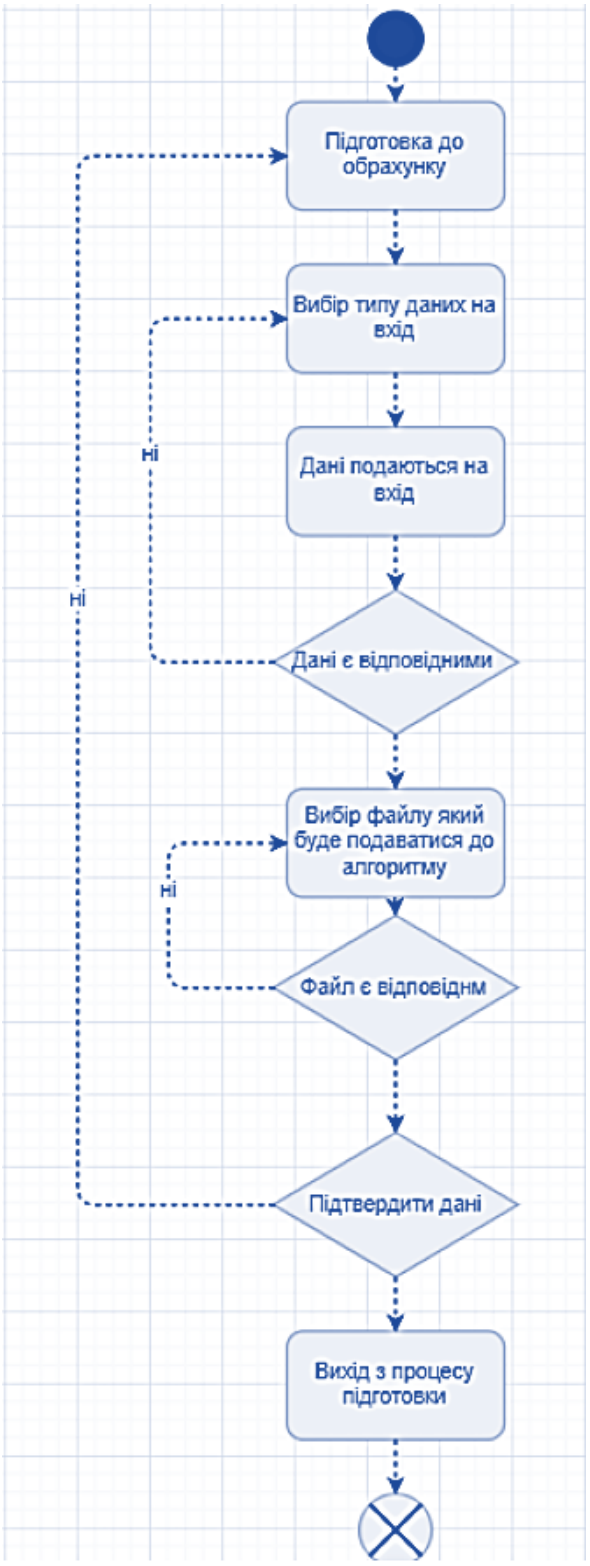


Рисунок 1.27. UML-діаграма введення та підготовки даних для обробки

Таке розбиття процесів дозволяє розробити її логіку на рівні програмного забезпечення з урахуванням обробки помилок, перевірки валідності даних та

забезпечення коректності всіх проміжних кроків. Програмна реалізація кодека Triple DES буде передбачати коплексну розробку окремих модулів, що охоплюють генерацію та розширення ключів, обробку даних із початковою перестановкою, реалізацію 16 раундів перетворень за допомогою мережі Фейстеля, а також організацію послідовного виконання трьох операцій DES, що забезпечує задовільну стійкість криптографічного захисту. Цей підхід забезпечує модульність та гнучкість майбутньої програми, яку буде реалізовано у Visual Studio мовою C#, що дозволить проводити детальне тестування кожного алгоритмічного компонента та легко інтегрувати додатковий функціонал у разі необхідності.

1.7 Підготовка засобів розробки програмного кодеку

Підготовка засобів розробки розпочинається з встановлення та налаштування останньої версії Visual Studio, яка забезпечує підтримку сучасних стандартів розробки, потужну систему налагодження коду, інтегровані інструменти для аналізу якості програмного забезпечення та зручність у створенні документованої архітектури проекту. Завдяки цьому середовищу розробник має можливість працювати з об’єктно-орієнтованою парадигмою, розбивати систему на окремі модульні компоненти, що суттєво полегшує підтримку та подальший розвиток програмного коду.

Одним із ключових аспектів підготовки є створення базової структури проектного рішення. Для цього формується рішення, яке містить окремі проекти для реалізації бізнес-логіки, побудови інтерфейсу користувача та проведення модульного тестування. Такий підхід забезпечує чітке розділення функціональних зон, дозволяючи ізолювати основні алгоритмічні компоненти – наприклад, модулі генерації ключів, виконання початкової перестановки даних, реалізації раундів алгоритму DES та організації послідовного шифрування і дешифрування відповідно до схеми “шифрування–дешифрування–шифрування”. Крім того, налаштування системи контролю версій із використанням Git сприятиме відстеженню змін, забезпеченню злагодженої командної роботи та покращенню процесів автоматичного тестування.

Не менш важливою є інтеграція допоміжних бібліотек та засобів, що

полегшують реалізацію криптографічних операцій. У процесі підготовки була проведена оцінка можливостей вбудованих криптографічних бібліотек платформи .NET, що дозволяють використовувати перевірені реалізації основних операцій шифрування й дешифрування, а також забезпечують додатковий рівень безпеки.

Для реалізації криптографічного кодеку Triple DES обрана платформа розробки – Microsoft Visual Studio 2022 із застосуванням мови програмування C# та технології Windows Forms. Використання Visual Studio 2022 забезпечує сучасне інтегроване середовище розробки, яке включає потужний інтерфейс, зручні засоби налагодження та управління проектом, а також механізм менеджера пакетів NuGet для простого інтегрування додаткових бібліотек. Як ілюстрація цього процесу, на Рисунку 1.28 наведено скріншот, що демонструє етап розробки візуального інтерфейсу застосунку кодеку за допомогою Windows Forms.

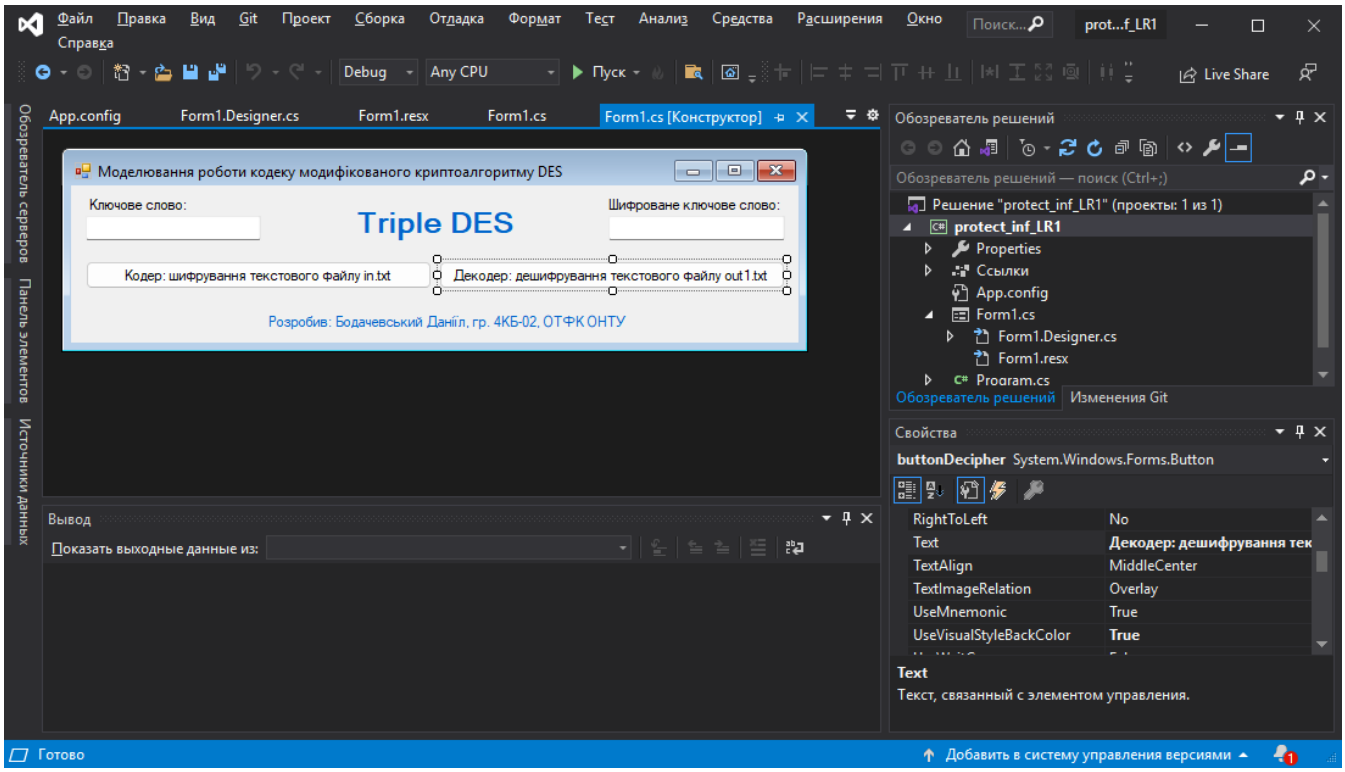


Рисунок 1.28. Інтерфейс Microsoft Visual Studio 2022 C# у режимі Windows Forms

При створенні програмного застосунку кодеку використовуються засоби мовою C#, що дозволяють реалізувати об’єктно-орієнтоване програмування з високим рівнем абстракції. Мова C# є простою у використанні, проте водночас надає повноцінну функціональність, необхідну для розробки великих і складних проектів. Вона є сучасним спадкоємцем мов C++ і Java, забезпечуючи строгий

контроль типів, підтримку поліморфізму, наслідування, перевантаження операторів, інкапсуляцію, обробку виключень і багато інших можливостей, що дозволяють писати чистий, модульний та легкий для подальшого супроводження код. Навіть стандартні типи даних, такі як гнучкі масиви, списки і словники, реалізовані на високому рівні, що значно скорочує час розробки в порівнянні з мовами нижчого рівня.

Windows Forms є технологією, яка дозволяє розробляти інтелектуальних клієнтів – застосунки з повноцінним графічним інтерфейсом, простими в розгортанні та оновленні. Ця технологія надає розробнику широкий набір елементів управління, включаючи текстові поля, кнопки, списки, меню та інші компоненти, що спрощують взаємодію користувача із системою. При цьому елементи управління можуть бути розміщені на формі інтуїтивно за допомогою конструктора Visual Studio, що підтримує такі функції, як вирівнювання за допомогою ліній сітки та ліній прив'язки, що дозволяє створювати ефективні й ергономічні інтерфейси. За допомогою Windows Forms легко реалізувати принцип поділу застосунку на модулі, що дозволяє використовувати окремі компоненти в інших проектах, а також забезпечує повторне використання вже перевірених блоків логіки. Цей підхід значно полегшує адаптацію та розширення функціоналу створеного кодека.

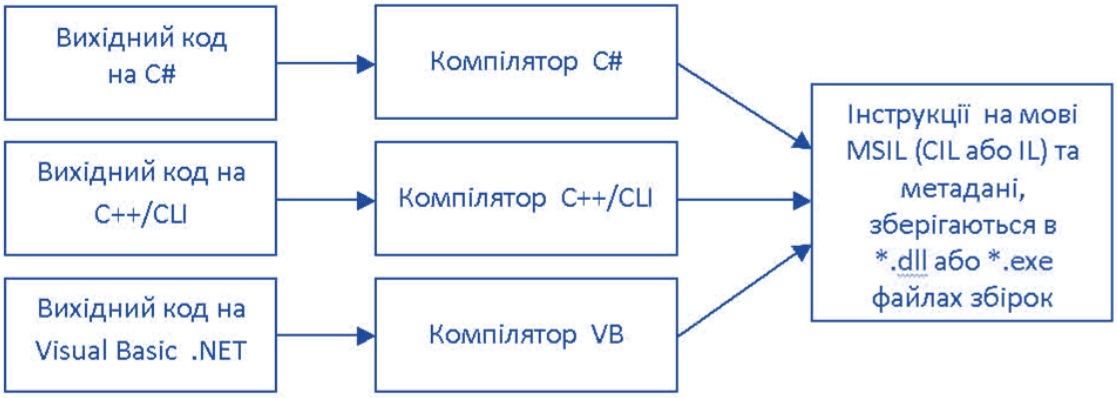


Рисунок 1.29. Схема перетворення вихідного коду в код мовою MSIL та утворення файлу збірки

Програми, написані мовою C#, виконуються на платформі .NET Framework, що включає середовище виконання Common Language Runtime (CLR). CLR відповідає за завантаження збірок, виконання JIT-компіляції проміжного коду (MSIL) у машинний код, а також забезпечує автоматичне збирання сміття, обробку

винятків і управління ресурсами, що значно підвищує стабільність роботи застосунку. На рис. 1.29 зображено процес перетворення вихідного коду на мову MSIL з подальшим утворенням файлу збірки, що демонструє тісну інтеграцію між вихідним кодом, бібліотеками .NET та середовищем CLR. Використання даної архітектури дозволяє досягти високого рівня безпеки і оптимізації виконання, що є критично важливим для розроблюваного криптографічного кодека.

1.8 Реалізація програмного застосунку кодеку криптоалгоритму Triple DES

Реалізація програмного застосунку кодеку криптоалгоритму Triple DES базується на розробці основної функції шифрування та дешифрування, що реалізована мовою C# у середовищі Microsoft Visual Studio 2022 із застосуванням технології Windows Forms. Основна функція шифрування, детальний код якої наведено у Додатку А, виконує послідовне застосування операцій шифрування та дешифрування відповідно до схеми «шифрування – дешифрування – шифрування», завдяки чому досягається високий рівень криптографічного захисту даних.

Рисунок 1.30. Інтерфейс програми-кодеку на базі алгоритму Triple DES

Головне вікно застосунку (рис. 1.30) містить ряд елементів управління, зокрема текстові поля для введення початкового тексту та ключового слова, поле для виведення зашифрованого ключа, а також кнопки для запуску процесів шифрування та дешифрування. Для шифрування початкового повідомлення, яке знаходиться у текстовому файлі in.txt (рис.1.31), користувач вводить ключове слово у відповідне текстове поле та натискає кнопку «Кодер: дешифрування текстового файлу in.txt». Після натискання цієї кнопки застосунок зчитує дані з файлу in.txt, конвертує їх у масив байтів і обробляє за допомогою основного методу шифрування,

який використовує алгоритм Triple DES. Результатом цього процесу є утворення шифротексту, який записується у файл out1.txt (рис. 1.32). Одночасно, для додаткового захисту, в спеціальне текстове поле «Шифроване ключове слово» записується зашифрований варіант введеного ключа.

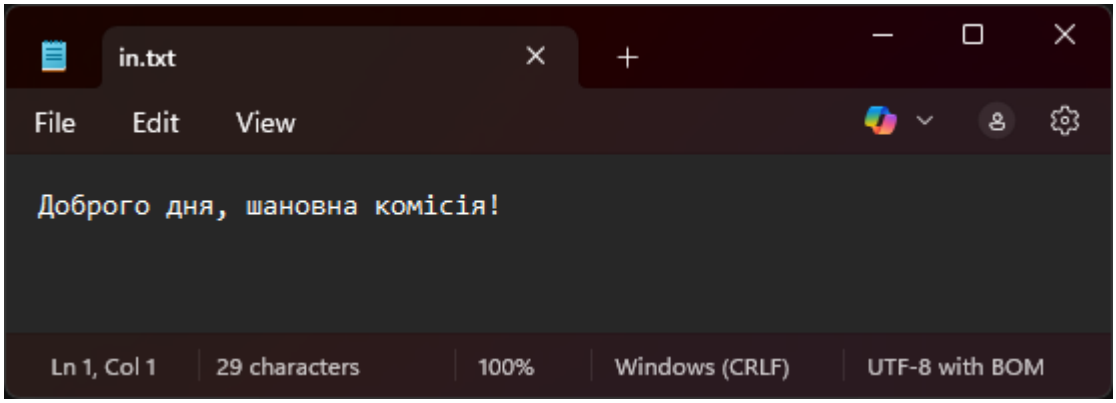


Рисунок 1.31. Приклад вмісту початкового текстового файлу in.txt

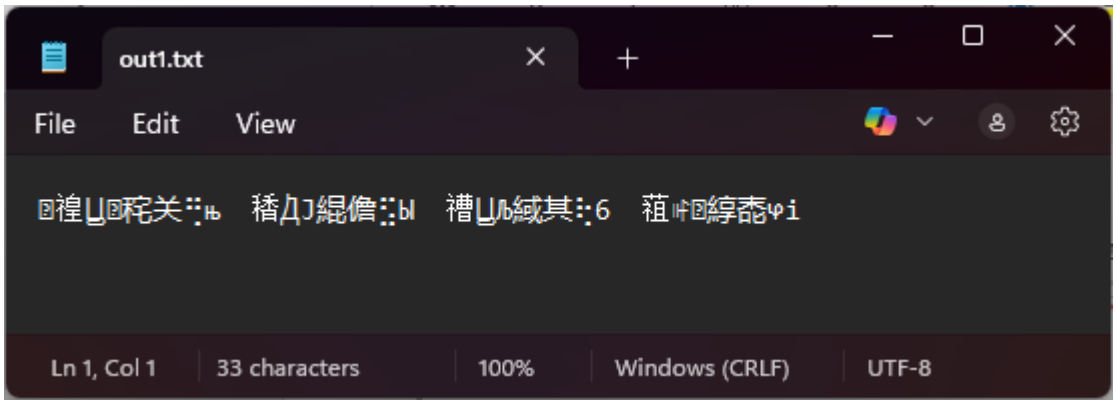


Рисунок 1.32. Приклад виведення результатів шифрування у файл out1.txt

Ключовим для реалізації є метод, що виконує шифрування за допомогою стандартного класу *TripleDESCryptoServiceProvider* з просторового імен *System.Security.Cryptography*. Нижче наведено фрагмент коду цього методу:

```
public static byte[] TripleDESEncrypt(byte[] plainData, byte[] key)
{
   // Генерація ключа відповідної довжини (24 байти)
   byte[] newKey = new byte[24];
   Array.Copy(key, newKey, Math.Min(key.Length, 24));
   using (TripleDESCryptoServiceProvider tdes = new TripleDESCryptoServiceProvider())
   {
      tdes.Key = newKey;
      tdes.Mode = CipherMode.ECB; // Встановлено режим шифрування за вимогою
      tdes.Padding = PaddingMode.PKCS7;
      ICryptoTransform encryptor = tdes.CreateEncryptor();
      return encryptor.TransformFinalBlock(plainData, 0, plainData.Length);
   }}
```

Даний фрагмент коду демонструє основну логіку шифрування даних: спершу виконується підготовка ключа (за необхідності доповнюється до 24 байтів), після чого створюється об’єкт *TripleDESCryptoServiceProvider*, в якого встановлюються режим роботи та схема заповнення. За допомогою методу *CreateEncryptor()* створюється об’єкт трансформатора, що перетворює вхідний масив байтів (початковий текст) у шифротекст.

При обробці натискання кнопки «Кодер: дешифрування текстового файлу in.txt» реалізовано подібну логіку:

```
private void btnCoder_Click(object sender, EventArgs e)
{
   // Перевірка наявності введеного ключа
   if (string.IsNullOrEmpty(txtKey.Text))
   {
      MessageBox.Show("Не введено ключ шифрування!", "Помилка",
      MessageBoxButtons.OK, MessageBoxIcon.Error);
      return;
   }
   try
   {
      // Зчитування вихідного тексту з файлу in.txt
      byte[] plainData = File.ReadAllBytes("in.txt");
      // Перетворення ключового слова в масив байтів (UTF8)
      byte[] keyData = Encoding.UTF8.GetBytes(txtKey.Text.Trim());
      // Шифрування даних за допомогою методу TripleDESEncrypt
      byte[] cipherData = TripleDESEncrypt(plainData, keyData);
      // Запис шифротексту у файл out1.txt
      File.WriteAllBytes("out1.txt", cipherData);
      // Для додаткового захисту шифрується сам ключ та виводиться в інтерфейсі
      txtEncryptedKey.Text = Convert.ToBase64String(TripleDESEncrypt(keyData, keyData));
      MessageBox.Show("Шифрування виконано успішно", "Інформація",
      MessageBoxButtons.OK, MessageBoxIcon.Information);
   }
   catch(Exception ex)
   {
   MessageBox.Show("Помилка шифрування: " + ex.Message, "Помилка",
   MessageBoxButtons.OK, MessageBoxIcon.Error);
   }
}
```

Цей код відповідає за обробку події кліку на кнопку шифрування. Він перевіряє наявність ключа, зчитує вихідні дані з файлу, перетворює їх у відповідний формат, викликає метод шифрування, після чого записує отриманий шифротекст у файл out1.txt. Додатково, у призначене текстове поле записується зашифрований варіант ключа для підвищення рівня захисту.

Аналогічно реалізовано і процес дешифрування. При натисканні кнопки «Декодер: шифрування текстового файлу out1.txt» виконується наступна логіка:

```
public static byte[] TripleDESDecrypt(byte[] cipherData, byte[] key)
{   // Генерація ключа відповідної довжини (24 байти)
     byte[] newKey = new byte[24];
   Array.Copy(key, newKey, Math.Min(key.Length, 24));
      using (TripleDESCryptoServiceProvider tdes = new TripleDESCryptoServiceProvider())
   {
       tdes.Key = newKey;
       tdes.Mode = CipherMode.ECB;
       tdes.Padding = PaddingMode.PKCS7;
       ICryptoTransform decryptor = tdes.CreateDecryptor();
       return decryptor.TransformFinalBlock(cipherData, 0, cipherData.Length);
   }
}
private void btnDecoder_Click(object sender, EventArgs e)
{
   // Перевірка наявності ключа дешифрування
   if (string.IsNullOrEmpty(txtKey.Text))
   {
       MessageBox.Show("Не введено ключ дешифрування!", "Помилка",
       MessageBoxButtons.OK, MessageBoxIcon.Error);
       return;
   }
   try
   {
       // Зчитування шифротексту з файлу out1.txt
       byte[] cipherData = File.ReadAllBytes("out1.txt");
       byte[] keyData = Encoding.UTF8.GetBytes(txtKey.Text.Trim());
      // Дешифрування даних за допомогою методу TripleDESDecrypt
       byte[] plainData = TripleDESDecrypt(cipherData, keyData);
       // Запис розшифрованого тексту у файл out2.txt
       File.WriteAllBytes("out2.txt", plainData);
       MessageBox.Show("Дешифрування виконано успішно", "Інформація",
       MessageBoxButtons.OK, MessageBoxIcon.Information);
   }
   catch(Exception ex)
   {
       MessageBox.Show("Помилка дешифрування: " + ex.Message, "Помилка",
       MessageBoxButtons.OK, MessageBoxIcon.Error);
   }
}
```

У цьому фрагменті виконуються аналогічні операції з обробки даних: зчитується шифротекст з файлу out1.txt, на основі введеного ключа створюється масив байтів та здійснюється операція дешифрування. Результат дешифрування записується у файл out2.txt, що гарантує відновлення початкового тексту відповідно до даних з in.txt (рис. 1.33).

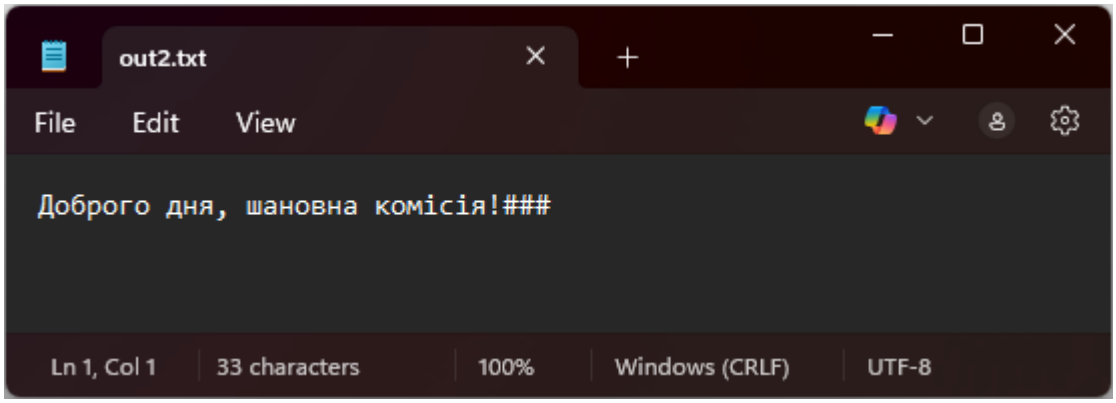


Рисунок 1.33. Приклад виведення результатів дешифрування у файл out2.txt

В разі відсутності введення ключа, відповідні перевірки у коді не дозволяють виконати операцію шифрування чи дешифрування, а система виводить повідомлення про помилку, як це продемонстровано на рис. 1.34.

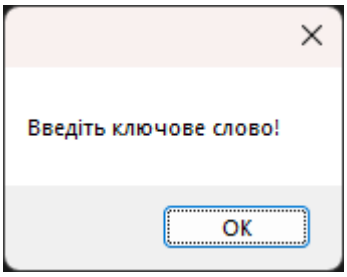


Рисунок 1.34. Виведення повідомлення у разі відсутності ключового слова у полі

Реалізація програмного застосунку кодеку Triple DES включає інтеграцію функціональних модулів, що забезпечують читання та запис файлів, перетворення вхідних даних у необхідний формат, основну логіку шифрування/дешифрування із використанням класу *TripleDESCryptoServiceProvider* та інтуїтивно зрозумілий графічний інтерфейс для взаємодії користувача. Завдяки використанню засобів Visual Studio 2022 і мови C#, система є модульною, легко підтримуваною і готовою до подальшої розширеної інтеграції у більші програмні рішення.

За наведеними вище скріншотами (рис.1.30 - рис.1.33) видно, що зашифровані/дешифровані розробленим кодеком повідомлення повністю співпадають з результатами модулювання у програмі Cryptool2, що свідчить про правильну роботу реалізованого застосунку.

2 ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

У ході виконання даного дипломного проєкту було проведено глибоке дослідження роботи модифікованих криптографічних алгоритмів DES, а також здійснено практичну реалізацію кодека, заснованого на алгоритмі Triple DES. Завдяки моделюванню в середовищі CrypTool2 вдалося ретельно проаналізувати процеси шифрування та розшифрування, що сприяло усуненню можливих недоліків і вдосконаленню логічної структури алгоритму. Отримані результати стали основою для створення алгоритмічних модулів майбутнього програмного продукту, що забезпечують модульну організацію, зручність тестування і перспективи розширення функціоналу.

Ефективність програмного забезпечення оцінюється як за його якістю, так і за продуктивністю процесу розробки. Якість ПЗ визначається кількома основними чинниками: зручністю використання, оптимальним розподілом ресурсів та відповідністю встановленим вимогам. З позиції користувача значущим аспектом є також фінансові витрати на розробку, які включають як трудові, так і матеріальні ресурси. У цьому розділі наведено розрахунок вартості створення розробленого програмного рішення.

2.2 Визначення трудомісткості розробки програмного забезпечення

Для оцінки обсягу програмного коду застосовується метод структурної аналогії, який дозволяє визначити кількість умовних машинних команд (у тисячах) на основі спеціалізованих каталогів схожих рішень.

Таблиця 2.1. Каталог аналогів

Найменування ПП	**Обсяг функції ПП – V_o, усл. машинних командах.**
1. ПП автоматизації засобів по каталогу	680 – 7000
2. ПП автоматизованих розрахунків	1300 – 8600
3. ПП оптимізації розрахунків	1300 – 4200

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт. Для нашого варіанта виділено сірим кольором.

Вибравши аналог ПЗ, що містить Vo в умовних машинних командах, трудомісткості визначати на основі табл.2.2

Таблиця.2.2. Норма часу

Обсяг ПЗ, тис.умов.машинних команд	**Норма часу, люд/год**
1.00	229
2.00	244
3.00	262

На підставі отриманого значення, по довіднику, визначається укрупнена норма часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПП, тобто в умовах комп'ютера, Кк=0,7÷0,8):

Тар = 229 × 0,8 = 183,2 люд.-год.

Трудомісткість програмного продукту визначається окремо для кожного етапу розробки, виходячи з показників трудомісткості відповідного аналога. При цьому враховують рівень складності розробки, ступінь інноваційності та частку використання стандартних модулів. Розрахунки проводяться згідно з наступними формулами:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \qquad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \qquad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \qquad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розробки (див. табл. 2.3.);

Кн – поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.4.);

Кт – поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.5)

Таблиця 2.3. Значення питомих коефіцієнтів трудомісткості стадії в загальній трудомісткості розробки ПП

Код стадії	**Ступінь новизни**		
	А	Б	В
ТЗ (L_1)	0,15	0,12	0,12
ТП (L_2)	0,16	0,15	0,11
РП (L_3)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	**Ступінь новизни**	**Значення $К_н$**
А	Принципово нові ПП	1,75 – 1,2
Б	ПП – розвиток визначеного параметричного ряду	1,0 – 0,8
В	ПП маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.5. Значення коефіцієнта ступеня використання в розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПП типовими програмами, %	**Значення $К_т$**
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=Т^а * L_1 * К_н = 183,2 * 0,12 * 0,7 = 15,38 \text{ (люд/годин)} \quad (2.4)$$

Трудомісткість розробки технічного проекту

$$Т_{тп}=Т^а * L_2 * К_н = 183,2 * 0,11 * 0,7 = 14,11 \text{ (люд/годин)} \quad (2.5)$$

Трудомісткість розробки робочого проекту

$$Т_{рп}=Т^а * L_3 * К_н * К_т = 183,2 * 0,61 * 0,7 * 0,6 = 46,94 \text{ (люд/годин)} \quad (2.6)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання $N_{тз}$= 2 (стр), розробка ТП $N_{тп}$=26 (стр), розробка робочого проекту $N_{рп}$=6(стр), пояснювальна записка відповідно $N_{пз}$=22(стр) Розрахунок зведений у таблицю 2.6.

Таблиця 2.6. Розрахунок трудомісткості ПП

Найменування етапів	**Розрахунок, годин**		
1.ТЗ	$T_{Ртз}$=15,38	$T_{кк}=0,7*N_{тз}$= 0,7*3=2,1	$T_{нк}=0,15*N_{тз}=0,15*3=0,45$
2.Розробка ТП	$T_{Ртп}$=14,11	$T_{кк}=0,7*N_{тп}=0,7*30=2$ 1,0	$T_{нк}=0,15*N_{тп}=0,15*30=4,5$
3.Розробка РП	$T_{Ррп}$= 46,94	$T_{кк}=0,7*N_{рп}=0,7*7=4,$ 9	$T_{нк}=0,15*N_{рп}=0,15*7=1,05$
4.Розробка ПЗ	$T_{пз}=1,5*N_{пз}=$ 1,5*17=25,5	$T_{кк}=0,7*N_{тз}=0,7*17=1$ 1,9	$T_{нк}=0,15*N_{пз}=0,15*17$ =2,55
Усього, в т.ч.:	150,38		
- на розробку	$\Sigma T_р$=101,93		
- контроль керівника		$\Sigma T_{кк}$=39,9	
- нормоконтроль			$\Sigma T_{нк}$=8,55

2.3 Розрахунок ціни програмного продукту

Для оцінки вартості програмного продукту розглядаються основна заробітна плата виконавців, матеріальні витрати та загальні витрати на розробку ПЗ. Детальний розрахунок основної заробітної плати наведено у таблиці 2.7. Згідно зі статтею 8 «Закону про Державний бюджет України на 2025» з 1 січня 2025 року встановлено мінімальну місячну заробітну плату у розмірі 8000 гривень, а також мінімальну погодинну тарифну ставку – 48,00 грн.

Таблиця 2.7. Розрахунок основної заробітної плати виконавців

Найменування робіт	**Трудомісткість робіт, години**	**Погодинна тарифна ставка, грн.**	**Розрахунок, грн.**
1.Розробка ПП	101,93	48,00	4892,64
2.Контроль керівника	39,9	120,00	4788,00
3.Нормоконтроль	8,55	110,00	940,50
Усього	-	-	ΣЗо= 10621,14

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок

зведемо в таблицю 2.8.

Таблиця 2.8. Розрахунок матеріальних витрат на розробку ПЗ

Найменування матеріальних витрат	**Тип, модель**	**Кількість**	**Ціна одиниці, грн.**	**Вартість, грн.**
Папір	Лист А4	57	5.0	285,0
Разом	-	-	-	$В_{м\,і}$=285,0
Транспортно – заготівельні Витрати (10%)				$Втр\_з = 0{,}1 \times Вм1$= 0,1*280=28,0
Усього				$В_м=В_{мі}+В_{тр\ з=}$ 308,00

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.9.

Таблиця 2.9. Розрахунок статей витрат планової собівартості

Стаття витрат	**Значення, грн.**	**Формула розрахунку**
1. Матеріали	313,5	$В_м$ (див. табл. 2.8.)
2. Основна заробітна плата	10621,14	$З_о$ (див. табл. 2.7.)
3.Додаткова заробітна плата	1062,11	$Зд = 0{,}1 \times Зо$ = 10621,14*0,1
4.Відрахування до єдиного фонду соціального внеску	2570,31	$Вє.с.в. = 0{,}22 \times (Зо + Зд)$ =0,22* (10621,14+1062,11)
5. Накладні витрати	4248,46	$Внак. = 0{,}4 \times Зо$ =0,4 * 10621,14
6. Повна собівартість	18815,52	$С_{пов}$= $В_м$ +$З_о$ + $З_д$ +Вє.с.в. + Внак.= 313,5+10621,14+1062,11+2570,31 +4248,46

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$П=(С_{пов}*Р)/100=(18815{,}52*15)/100=2822{,}33$ грн (2.7)

Де р – плановий рівень рентабельності (10-15%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$Ц_о = С_{пов}+П = 18815{,}52+2822{,}33 = 21637{,}85$ грн; (2.8)

Виходячи з отриманих даних, ціна реалізації розробленого програмного забезпечення становитиме:

$Ц_р=Ц_о+ПДВ = 21637{,}85 +21637{,}85 *0.2 = 25965{,}42$ грн; (2.9)

3 РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

Сучасне застосування комп'ютерних технологій відіграє важливу роль у розвитку криптографічних систем, сприяючи автоматизації складних процесів шифрування та дешифрування, оптимізації обробки даних і швидкому доступу до захищених інформаційних ресурсів. Це сприяє підвищенню ефективності криптографічних алгоритмів та посиленню інформаційної безпеки. Однак використання комп'ютерних систем також створює виклики, зокрема навантаження на обчислювальні ресурси та необхідність забезпечення високого рівня захисту даних.

У межах даного дослідження основна увага приділяється моделюванню роботи кодеку на основі модифікованого криптоалгоритму DES. Створення аналітичної моделі дозволяє провести детальну оцінку життєздатності, надійності та безпеки алгоритму шифрування, а також оптимізувати його структуру для ефективнішої обробки інформації. Користувацьке робоче середовище включає системний блок із програмним забезпеченням, розробленим спеціально для аналізу стійкості криптографічних систем та мереж. При цьому важливим залишається дотримання стандартних вимог інформаційної безпеки, що гарантує надійний захист даних.

Таким чином, створення програмної моделі модифікованого криптоалгоритму DES не тільки вдосконалює механізми шифрування, а й враховує різні фактори, що впливають на криптографічну стійкість у реальних умовах функціонування. Це сприяє розробці більш ефективних рішень у сфері інформаційної безпеки та підвищенню рівня захисту даних.

3.1 Аналіз небезпечних і шкідливих факторів, що впливають на користувача ПК

До основних критеріїв забезпечення гігієни робочого середовища належать інтенсивність освітлення, температура повітря, вологість, рівень шумового забруднення, ступінь вібраційного впливу, токсичність, загазованість. Крім цього, враховується дія електростатичного поля та вплив як неіонізуючих, так і іонізуючих електромагнітних випромінювань.

3.2 Гігієнічні вимоги до виробничого середовища

Державні санітарні норми, зокрема ДСАНПіН 3.3.2.007-98 «Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин», спрямовані на запобігання негативного впливу шкідливих чинників, що супроводжують роботу з візуальними дисплейними терміналами, на здоров’я працівників.

3.2.1 Вимоги до приміщення

Розміщення робочих місць із використанням ВДТ, ЕОМ і ПЕОМ заборонено у підвальних приміщеннях та на цокольних поверхах. Для кімнат, призначених для роботи з візуальними дисплейними терміналами, рекомендується орієнтувати вікна у напрямку півночі або північного сходу. На вікнах повинні бути встановлені регульовані жалюзі або штори, що дозволяють їх повністю закривати для забезпечення оптимальних умов освітлення.

Планувальні рішення будівель і приміщень, де розташовано відеодисплейні термінали, мають відповідати вимогам ДСАНПіН 3.3.2.007-98. Для робочого місця програміста передбачено мінімальну площу не менше 6 кв. м та об’єм приміщення не менше 20 куб. м. Крім того, стіни приміщень повинні бути пофарбовані матовою фарбою, а в приміщеннях з ВДТ обов’язково мають бути передбачені зони для відпочинку та психологічного розвантаження.

3.2.2 Освітлення

Для забезпечення належного освітлення приміщення, де працює програміст, застосовується комбінована система, що поєднує природне освітлення із додатковим штучним світлом. Загальне оздоблення простору виконується за допомогою газорозрядних ламп типу ЛД. Згідно з встановленими нормами, для робочого місця, на якому здійснюються високоточні операції (де мінімальний розмір об'єкта розрізнення становить 0,3–0,5 мм), необхідна освітленість рівномірно має досягати 300 лк. В цілому, ці вимоги щодо освітлення забезпечені.

3.2.3 Шум

У робочих приміщеннях основним джерелом шумового навантаження є звуки, що генеруються ПЕОМ. Крім того, значну частину шуму створюють джерела електромагнітного походження – це коливання компонентів електромеханічних пристроїв під впливом змінних магнітних полів. До того ж, в приміщеннях виникає структурний шум, який випромінюють поверхні конструктивних елементів (стіни, перекриття, перегородки) у звуковому спектрі частот. Для зниження або усунення негативного впливу шуму доцільно ізолювати робочі зони, розташовуючи їх у частинах будівлі, що знаходяться в глибині та ведуть своїми вікнами у двір – таким чином мінімізується вплив міського шуму. Крім цього, необхідно регулярно перевіряти герметичність корпусів комп'ютерної техніки та своєчасно здійснювати заміну вентиляторів охолодження.

3.3 Вимоги до організації робочого місця працівника

Конструкція робочого місця користувача комп'ютера, з урахуванням розташування сидіння, засобів керування та засобу відображення інформації, розроблена згідно з антропометричними, фізіологічними та психологічними вимогами, а також відповідно до специфіки виконуваної роботи. Робоче меблеве обладнання повинно бути оснащене можливістю індивідуального регулювання, що дозволить адаптувати його під зріст кожного користувача й підтримувати оптимальну, зручну поставу. Робочий стіл рекомендовано обробляти матовим покриттям, що сприяє зменшенню небажаних відблисків. > > Розміщення дисплея організовано таким чином, щоб його верхня межа відповідала рівню очей, а відстань до екрану становила приблизно 70 см – що повністю входить у допустимий інтервал від 60 до 90 см. Частота мерехтіння екрану fмер дорівнює 100 Гц, що значно перевищує мінімальне рекомендоване значення у 70 Гц. Крім цього, робоче місце розташовано перпендикулярно до віконних прорізів, що дозволяє уникнути прямого та відбитого світлового мерехтіння від вікон та джерел штучного освітлення.

3.4 Мікроклімат

Показники мікроклімату, складу іонів у повітрі, а також рівень шкідливих речовин у робочих зонах, де використовуються ПК, мають відповідати вимогам ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень».

Для підтримки нормативних значень мікроклімату та забезпечення оптимального співвідношення позитивних і негативних іонів слід передбачити установку пристроїв зволоження, штучної іонізації або кондиціювання повітря. Крім того, рівні інфрачервоного випромінювання не повинні перевищувати встановлених нормативних меж згідно з ГОСТ 12.1.005. Також вміст озону в робочій зоні не має перевищувати 0,1 мг/м³, оксидів азоту – 5 мг/м³, а концентрація пилу повинна залишатися в межах 4 мг/м³.

3.5 Електробезпека

Приміщення, де використовуються імпульсні джерела живлення згідно з ОНТП24-86 і ПУЕ-87, віднесено до категорії об'єктів, де ризик ураження персоналу електричним струмом не є підвищеним. Це пояснюється тим, що відносна вологість повітря не перевищує 75%, температура залишається нижчою за 35°С, а хімічно агресивні середовища відсутні. Електроживлення обладнання організовано від двофазної мережі з заземленою нейтраллю, при напрузі 220 В і частоті 50 Гц, із застосуванням автоматичних пристроїв токового захисту.

В приміщенні обов’язково має бути встановлена схема заземлення. Ураження електричним струмом може виникнути у випадках: 1) при контакті з відкритими струмоведучими елементами; 2) при торканні неструмоведучих частин обладнання, які, через порушення ізоляції або інші причини, опинилися під напругою.

Відповідно до вимог ГОСТ-12.2.007.0-75 устаткування (за винятком ЕОМ II класу) відноситься до I класу та оснащене робочою ізоляцією згідно з ГОСТ 12.1.009-76. Підключення обладнання здійснено згідно з нормативами ПБЕ та ПУЕ, тому додаткових заходів щодо електробезпеки не вимагається.

3.6 Пожежна безпека

Робоче приміщення, що відповідає вимогам ПБЕ та ОНТП 24–86 у сфері вибухово-пожежної безпеки, класифікується як об’єкт категорії «В».

Відповідно до ПУЕ, для зниження ризику виникнення пожежі необхідно забезпечити комплекс заходів, зокрема: ретельну ізоляцію всіх струмоведучих проводів, що підключені до робочих місць, регулярний огляд та перевірку стану їх ізоляції, а також суворе дотримання норм безпечної експлуатації обладнання.

Для гасіння пожеж на робочому місці користувача ПК застосовують як вуглекислотні, так і порошкові вогнегасники.

– Вуглекислотні вогнегасники випускаються у варіанті ручних пристроїв (наприклад, ВВК-5);

– Порошкові вогнегасники представлені моделями ВП-2, ВП-5, ВП-10 та іншими

Рисунок 3.1. Засоби пожежогасіння

З метою своєчасного оповіщення, на дільниці необхідно встановити протипожежну сигналізацію. Проходи та запасні виходи повинні бути вільними. Пожежний щит повинен розміщуватись в доступному місці та містити первинні засоби пожежогасіння (вогнегасник, лопату, відро, простирадло, ящик з піском)

ВИСНОВКИ

У результаті виконання даного дипломного проекту було проведено комплексне дослідження роботи модифікованих криптоалгоритмів DES і виконано практичну реалізацію криптографічного кодеку на основі алгоритму Triple DES. Робота включала аналіз базового алгоритму DES, його модифікацію із застосуванням потрійного шифрування для підсилення захисту даних, моделювання роботи алгоритму у середовищі CrypTool2, а також розробку програмного застосунку з використанням Visual Studio 2022, мови C# та технології Windows Forms.

Під час дослідження було виявлено, що базовий алгоритм DES, незважаючи на свою історичну значимість та простоту реалізації, має явні обмеження через короткий розмір ключа та вразливості до сучасних криптоатак. Проте, застосування потрійного алгоритму DES (Triple DES) дозволяє суттєво підвищити рівень криптографічного захисту завдяки використанню трьох незалежних ключів і послідовному виконанню операцій шифрування та дешифрування за схемою EDE.

Дипломний проект демонструє комплексний підхід до розробки криптографічних засобів захисту інформації: від теоретичного аналізу і моделювання алгоритму до практичної реалізації програмного застосунку.

Моделювання роботи алгоритму у CrypTool2 надало можливість детально прослідкувати за кожним етапом шифрування і дешифрування, що дозволило виявити та усунути можливі неточності й покращити загальну логіку алгоритму. Результати моделювання були використані як основа для створення алгоритмічних модулів майбутнього програмного застосунку, що забезпечують модульність, легке тестування та розширюваність системи.

Отримані результати можуть бути використані не лише для подальшого розвитку криптографічних систем, але й як навчальний матеріал для дослідження сучасних підходів до розробки безпечних програмних рішень. Розроблений кодек відповідає вимогам до сучасного рівня безпеки та надійності, що є особливо актуальним у сучасних умовах зростання вимог до захисту інформації.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Сидоренко В. А., Черниш Ю. І. Комп’ютерна криптографія. – Київ: Наукова думка, 2022. – 320 с.
2. Бойко В. О. Сучасні методи криптографічного захисту інформації. – Харків: НТУУ «ХПІ», 2020. – 256 с.
3. Шевченко П. О. Розробка програмного забезпечення на платформі .NET: Windows Forms. – Львів: ФЕМІ, 2023. – 210 с.
4. Ковальчук М. І. Програмування на C#. Практичний посібник. – Київ: Видавничий портал, 2021. – 352 с.
5. Іваненко О. С. Основи сучасної криптографії. – Одеса: Одеський науковий центр, 2021. – 288 с.
6. Петренко Л. М. Технології захисту інформації в інформаційних системах. – Київ: Центр учбової літератури, 2020. – 312 с.
7. Довженко Т. В. Криптографія: теорія та практика. – Харків: Харківський технічний університет, 2022. – 350 с.
8. Кириленко А. І. Інформаційна безпека в умовах цифрової трансформації. – Київ: Освіта України, 2019. – 275 с.
9. Марченко В. Ф. Методи захисту інформації та криптографічні системи. – Львів: Львівський політехнічний інститут, 2020. – 300 с.
10. Смірнов І. О. Основи комп’ютерної безпеки. – Київ: Університет «КПІ», 2021. – 290 с.
11. Романенко В. П. Безпека інформаційних технологій: підручник. – Київ: Академія, 2022. – 330 с.
12. Лазаренко Ю. В. Програмування під .NET Framework: сучасні підходи. – Харків: Фоліант, 2021. – 310 с.
13. Міністерство цифрової трансформації України. Сучасні криптографічні алгоритми: DES та 3DES [Електронний ресурс]. – Режим доступу: https://www.mct.gov.ua (20.04.2025).
14. Електронна бібліотека України. Інформаційна безпека та криптографія [Електронний ресурс]. – Режим доступу: https://www.elib.gov.ua (20.04.2025).

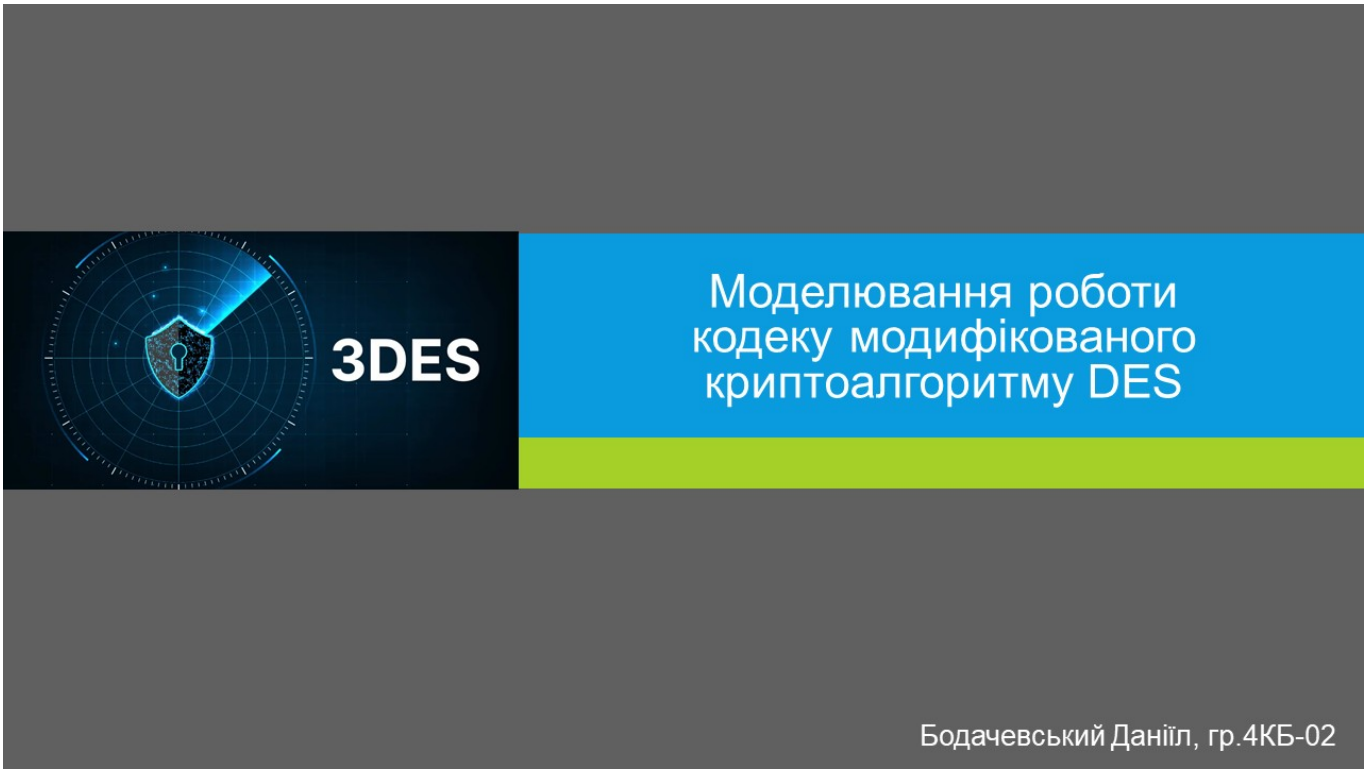
ДОДАТОК А. Фрагмент коду мовою C# методів GetKeyBytes, TripleDESEncrypt, TripleDESDecrypt кодеку Triple DES

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;
using System.Windows.Forms;
using System.Diagnostics;

namespace TripleDESCodec
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // <summary>
        /// Метод для отримання ключа довжиною 24 байти.
        // Triple DES вимагає ключ 192-біт (24 байти).
        // Якщо введений ключ коротший – доповнюється нулями, а якщо довший
        // – обрізається.
        // </summary>
        // <param name="key">Вхідний текст ключа</param>
        // <returns>Масив байтів довжиною 24 байти</returns>
        private byte[] GetKeyBytes(string key)
        {
            byte[] keyBytes = Encoding.UTF8.GetBytes(key);
            if (keyBytes.Length < 24)
            {
                // Доповнюємо до 24 байт (192 біт)
                Array.Resize(ref keyBytes, 24);
            }
            else if (keyBytes.Length > 24)
            {
                // Якщо ключ надто довгий, обмежуємо до 24 байт
                Array.Resize(ref keyBytes, 24);
            }
            return keyBytes;
        }

        // <summary>
        // Функція шифрування за алгоритмом Triple DES
        // Використовується клас TripleDESCryptoServiceProvider із простим
        //режимом шифрування (ECB) та заповненням PKCS7.
        // </summary>
        // <param name="plainData">Вхідний масив байтів (поширено
        // представлений текстовий файл)</param>
        // <param name="key">Масив байтів, отриманий із введеного слова за
        // допомогою GetKeyBytes</param>
        // <returns>Зашифрований масив байтів</returns>
        private byte[] TripleDESEncrypt(byte[] plainData, byte[] key)
        {
            using (TripleDESCryptoServiceProvider tdes = new
                   TripleDESCryptoServiceProvider())
            {
                tdes.Key = key;
                tdes.Mode = CipherMode.ECB; // Для демонстрації
        // використовується ECB; для реальних застосунків рекомендовано
        // використовувати CBC.
                tdes.Padding = PaddingMode.PKCS7;
                ICryptoTransform encryptor = tdes.CreateEncryptor();
                return encryptor.TransformFinalBlock(plainData, 0,
```

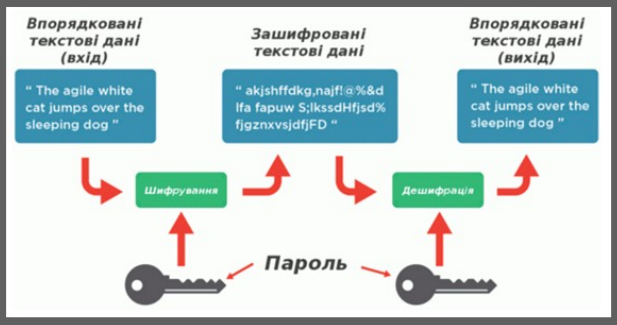
```
    plainData.Length);
    }
}

// <summary>
// Функція дешифрування за алгоритмом Triple DES
// Використовується аналогічно алгоритм шифрування із встановленими
// параметрами Mode та Padding.
// </summary>
// <param name="cipherData">Зашифрований масив байтів</param>
// <param name="key">Масив байтів ключа</param>
// <returns>Розшифрований масив байтів</returns>
private byte[] TripleDESDecrypt(byte[] cipherData, byte[] key)
{
    using (TripleDESCryptoServiceProvider tdes = new
// TripleDESCryptoServiceProvider())
    {
        tdes.Key = key;
        tdes.Mode = CipherMode.ECB;
        tdes.Padding = PaddingMode.PKCS7;
        ICryptoTransform decryptor = tdes.CreateDecryptor();
        return decryptor.TransformFinalBlock(cipherData, 0,
        cipherData.Length);
    }
}
// <summary>
// Обробник події натискання кнопки для шифрування:
// - Зчитує вихідний текст з файлу in.txt;
// - Отримує ключ із текстового поля та перетворює його у масив
// байтів необхідної довжини;
// - Виконує шифрування за допомогою функції TripleDESEncrypt;
// - Записує зашифрований текст у файл out1.txt;
// - Відображає зашифрований вигляд ключа для додаткового захисту.
// </summary>
private void buttonEncrypt_Click(object sender, EventArgs e)
{
    string keyText = textBoxKey.Text.Trim();
    if (string.IsNullOrEmpty(keyText))
    {
        MessageBox.Show("Введіть ключове слово для шифрування!",
        "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }

    try
    {
        // Зчитування вихідного тексту з файлу in.txt
        byte[] plainData = File.ReadAllBytes("in.txt");
        byte[] keyBytes = GetKeyBytes(keyText);

        // Виконання шифрування
        byte[] cipherData = TripleDESEncrypt(plainData, keyBytes);
        // Запис зашифрованого тексту в out1.txt
        File.WriteAllBytes("out1.txt", cipherData);

        // Додатково шифруємо сам ключ для відображення в
        // спеціальному полі інтерфейсу
        byte[] encryptedKey = TripleDESEncrypt(keyBytes, keyBytes);
        textBoxEncryptedKey.Text =
        Convert.ToBase64String(encryptedKey);

        MessageBox.Show("Шифрування виконано успішно!", "Інформація",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        Process.Start("out1.txt");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Помилка при шифруванні: " + ex.Message,
```

```
                    "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }

        // <summary>
        // Обробник події натискання кнопки для дешифрування:
        // - Зчитує шифротекст з файлу out1.txt;
        // - Перетворює введений ключ у відповідний масив байтів;
        // - Виконує дешифрування за допомогою функції TripleDESDecrypt;
        // - Записує відновлений текст у файл out2.txt.
        // </summary>
        private void buttonDecrypt_Click(object sender, EventArgs e)
        {
            string keyText = textBoxKey.Text.Trim();
            if (string.IsNullOrEmpty(keyText))
            {
                MessageBox.Show("Введіть ключове слово для дешифрування!",
                "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }

            try
            {
                // Зчитування зашифрованого тексту з файлу out1.txt
                byte[] cipherData = File.ReadAllBytes("out1.txt");
                byte[] keyBytes = GetKeyBytes(keyText);

                // Виконання дешифрування
                byte[] plainData = TripleDESDecrypt(cipherData, keyBytes);
                // Запис розшифрованого тексту в out2.txt
                File.WriteAllBytes("out2.txt", plainData);

                MessageBox.Show("Дешифрування виконано успішно!",
             "Інформація", MessageBoxButtons.OK, MessageBoxIcon.Information);
                Process.Start("out2.txt");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка при дешифруванні: " + ex.Message,
                "Помилка", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
    }
}
```

ДОДАТОК Б. Слайди мультимедійної презентації



Принцип симетричного шифрування у криптосистемі



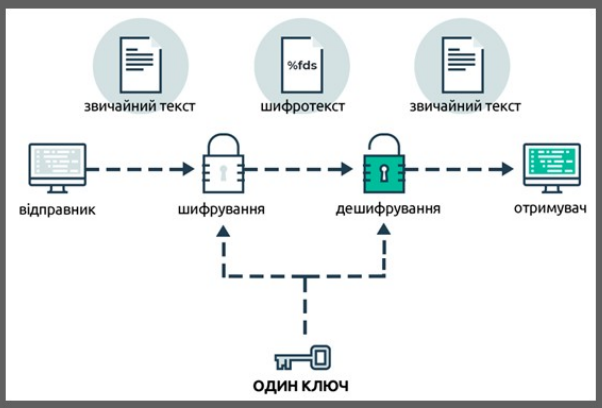
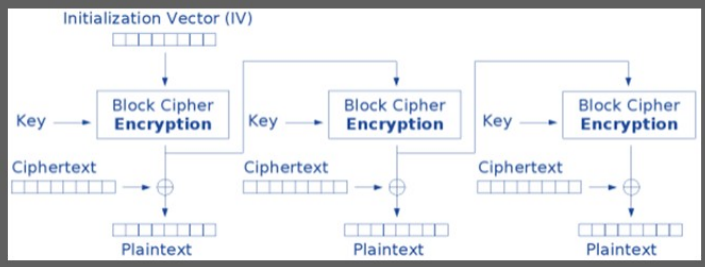
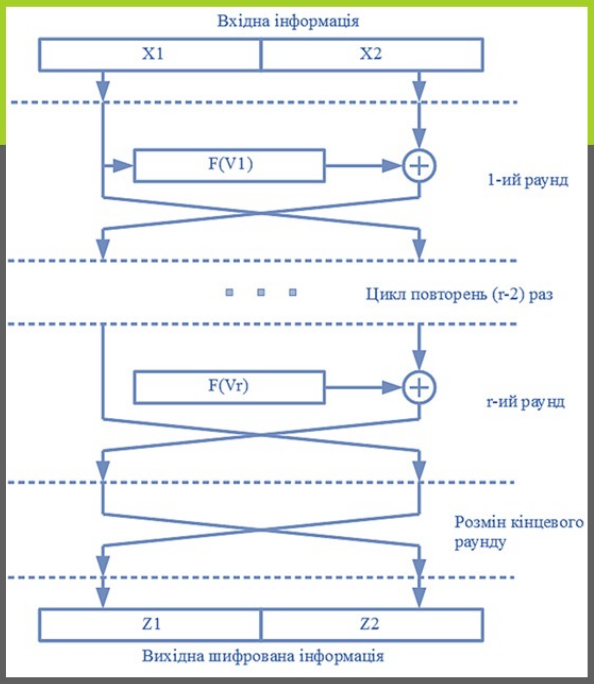
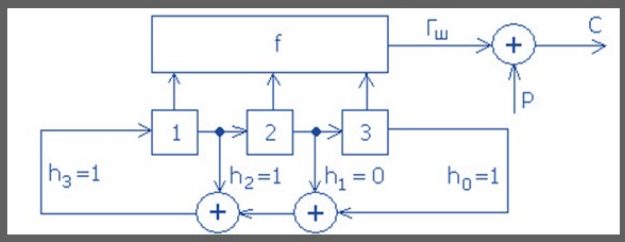


Схема блокового шифрування у криптосистемі



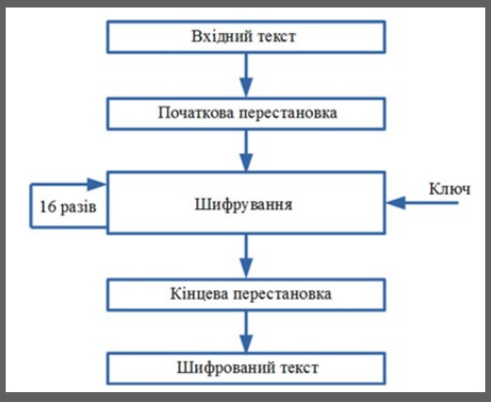
Базова структура мережі Фейстеля при шифруванні

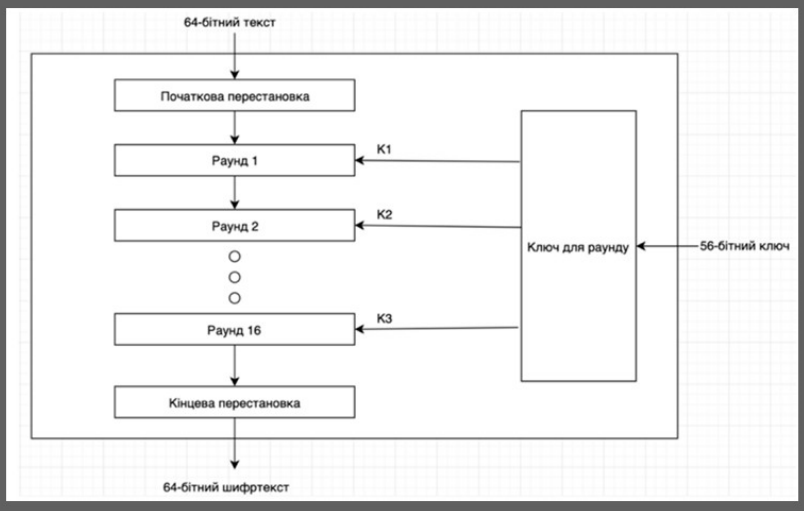


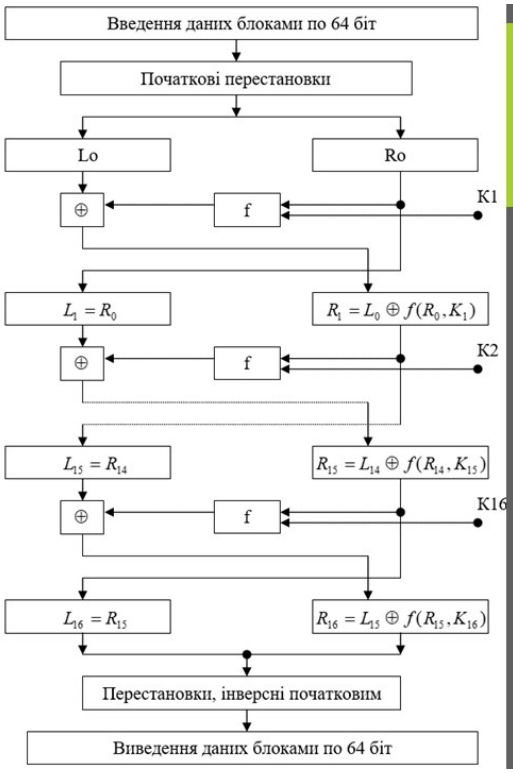


Застосування операції XOR у шифруванні

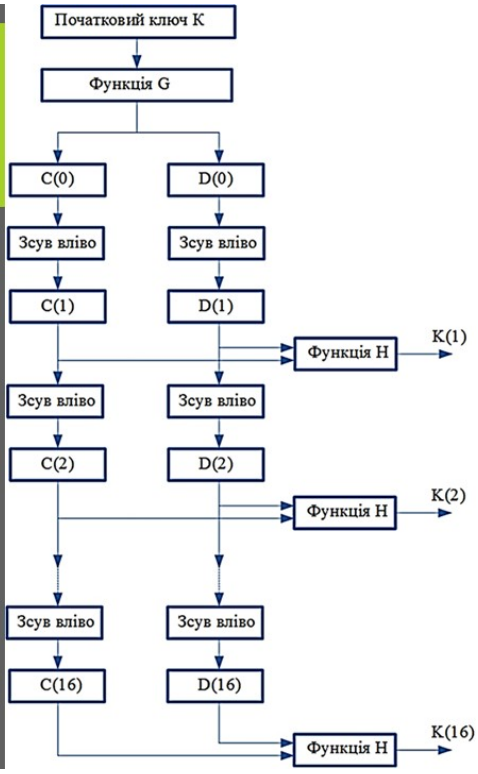
Структурні схеми роботи криптоалгоритму DES



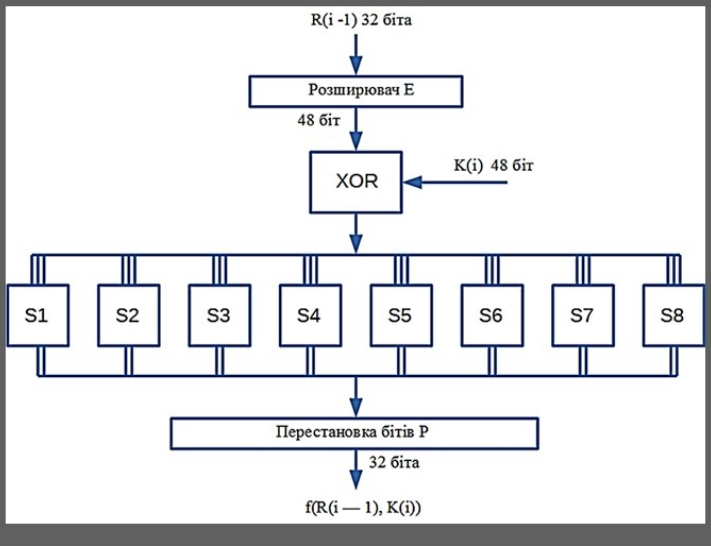




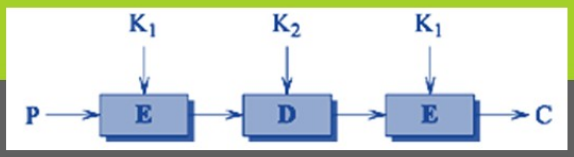
Блок-схема алгоритму шифрування DES та генерування ключів



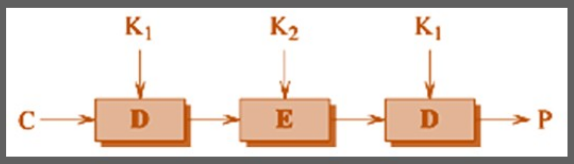
Обчислення функції шифрування f у алгоритмі DES



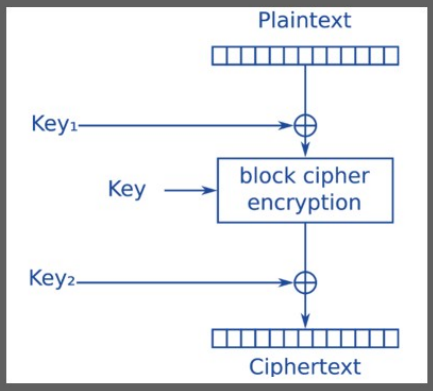
Принцип шифрування криптоалгоритмом DES-X



Принцип шифрування криптоалгоритмом 3DES



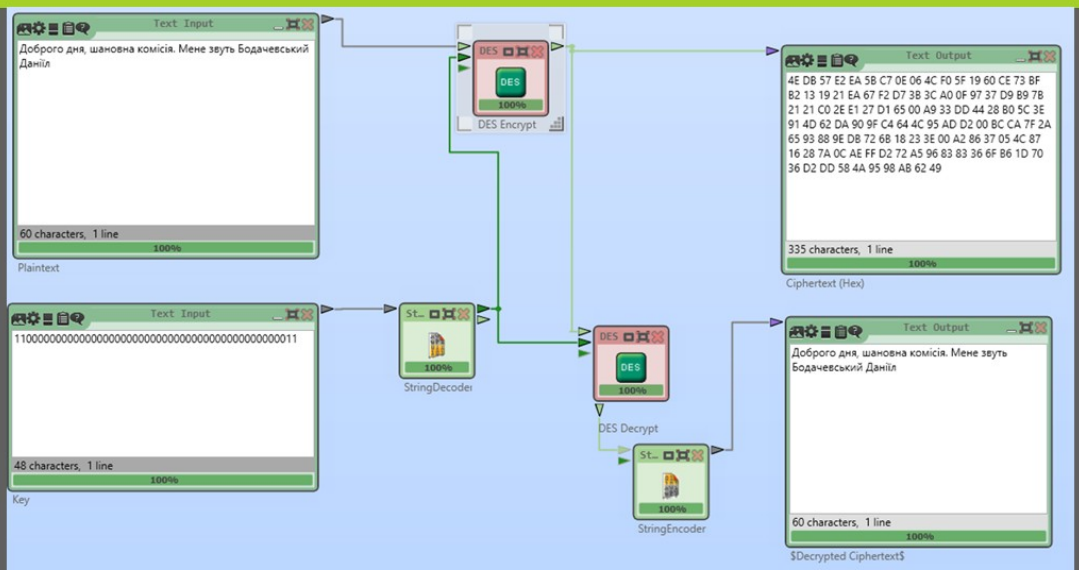
Принцип дешифрування криптоалгоритмом 3DES



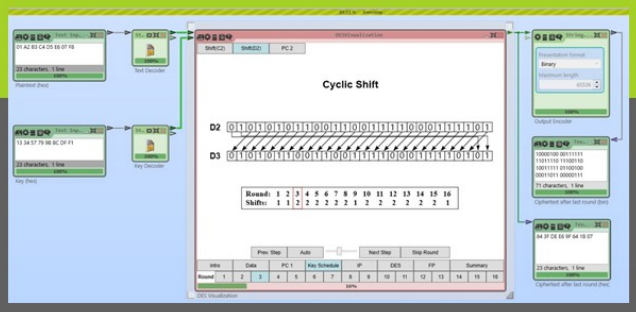
Порівняння DES (3DES, DES-X) та AES

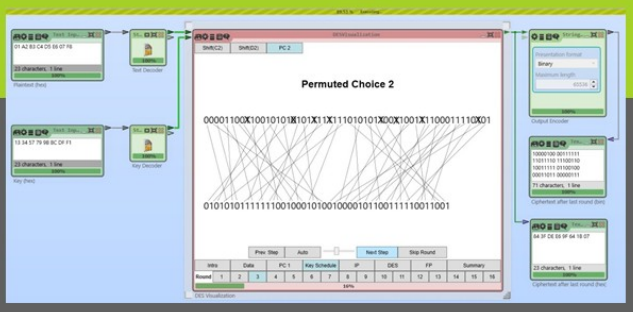
Алгоритм	Довжина ключа (біт)	Розмір блоку (біт)	Кількість раундів	Криптостійкість	Швидкість	Інші характеристики
DES	56 (загалом 64, але 8 біт – парність)	64	16	Низька. За сучасними стандартами ключ занадто короткий, що робить його вразливим до перебору та спеціалізованих атак.	Середня (працює непогано в апаратних рішеннях, але програмна реалізація вже застаріла).	Використовує мережу Фейстеля, має застосування S-блоків і перестановок; наразі не рекомендується для захисту критично важливої інформації.
3DES	112 (двоключова версія) або 168 (три незалежних ключа)	64	48 (3×16)	Висока. Значно підсилює безпеку порівняно з DES завдяки тричі послідовному застосуванню алгоритму, проте залишається обмеження 64-бітового блоку.	Низька. Виконується тричі, що уповільнює операції, тому використовується переважно в системах, де сумісність важливіша, ніж продуктивність.	Сумісний із існуючими DES-системами; у двоключовій версії піддається meet-in-the-middle атакам, повна (168-бітова) версія забезпечує максимальний рівень безпеки, але за рахунок продуктивності.
DES-X	Орієнтовно ~120 (за рахунок додаткового «whitening» ключа)	64	16 (основний DES + додаткові операції XOR на початку та в кінці)	Середня–середньо-висока. Розширює ефективну довжину ключа, що ускладнює прямий перебір, проте не вирішує всіх структурних недоліків DES.	Подібна до DES, оскільки додаткові операції XOR практично не ускладнюють процес; швидкість залишається високою.	Розширює захист шляхом перед-і після-обробки даних (whitening), що покращує стійкість до переборів, але фундаментально базується на структурі DES.
AES	128, 192 або 256	128	10, 12 або 14 (залежно від довжини ключа)	Дуже висока. Сучасний стандарт шифрування з високою стійкістю до широкого спектру атак, включаючи диференціальний і лінійний аналіз.	Висока. Швидка як в апаратних, так і в програмних реалізаціях із підтримкою апаратного прискорення (наприклад, інструкції AES-NI).	Працює за принципами заміщення-перестановки; широко впроваджений у глобальних системах; гнучкість у виборі розміру ключа дозволяє адаптувати рівень безпеки до конкретних вимог.

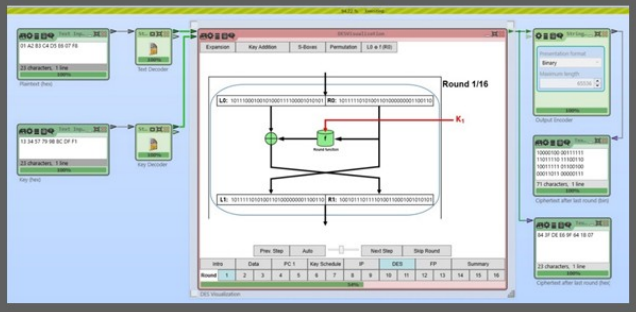
Результат моделювання у CrypTool2 за алгоритмом Triple DES

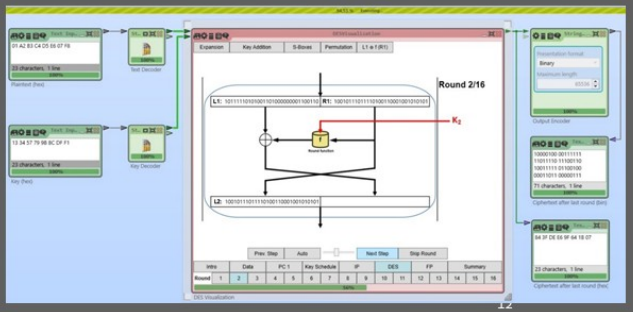


Кроки моделювання алгоритму DES при шифруванні даних

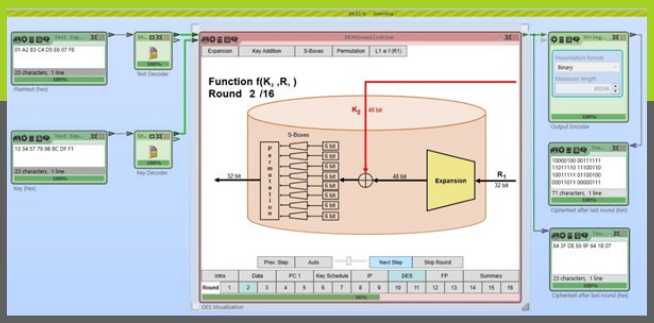


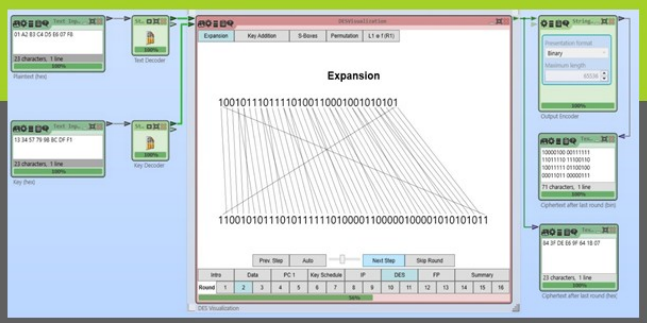


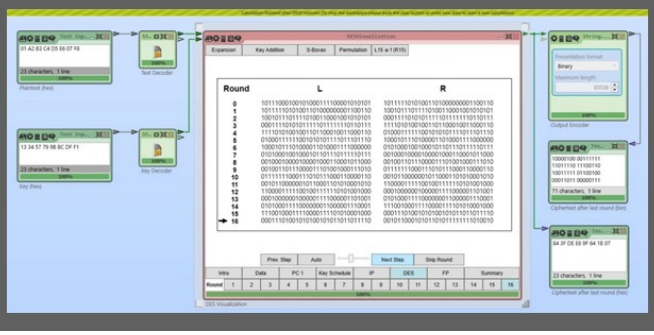


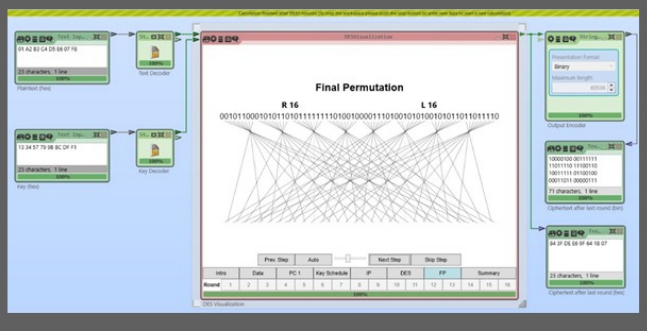


Кроки моделювання алгоритму DES при шифруванні даних

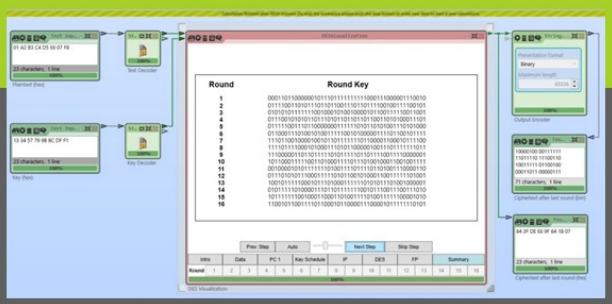


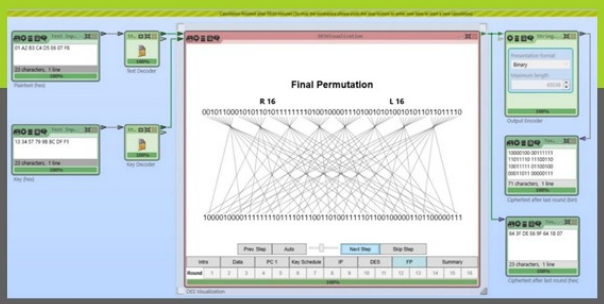


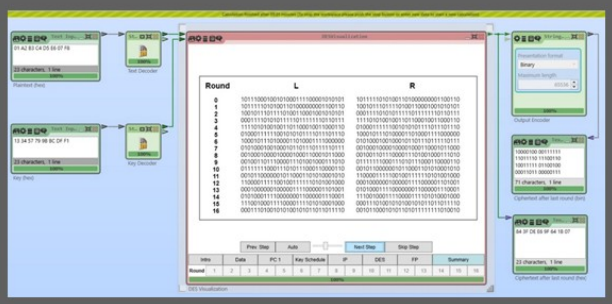


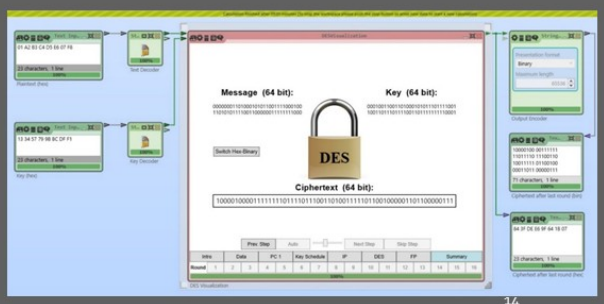


Кроки моделювання алгоритму DES при шифруванні даних

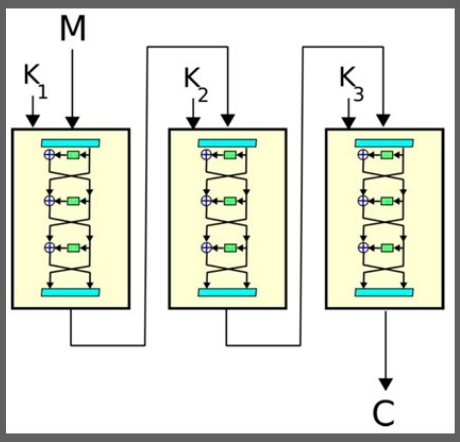




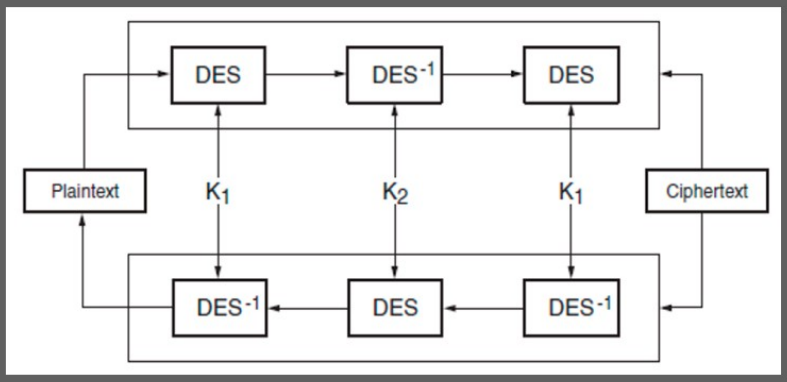




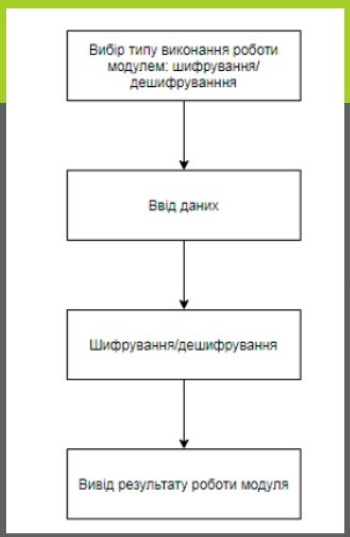
Модифікація алгоритму DES – Triple DES (3DES)



Структурна схема роботи криптоалгоритму Triple DES

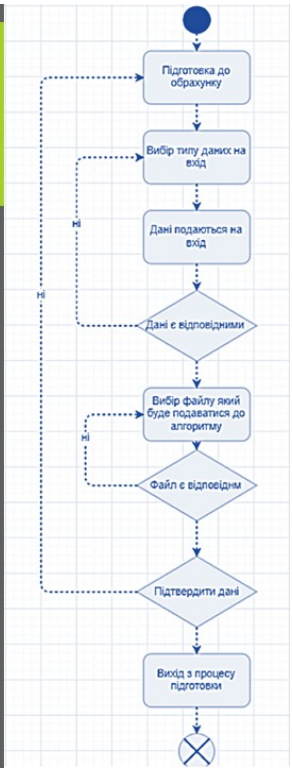


Блок-схема алгоритму DES

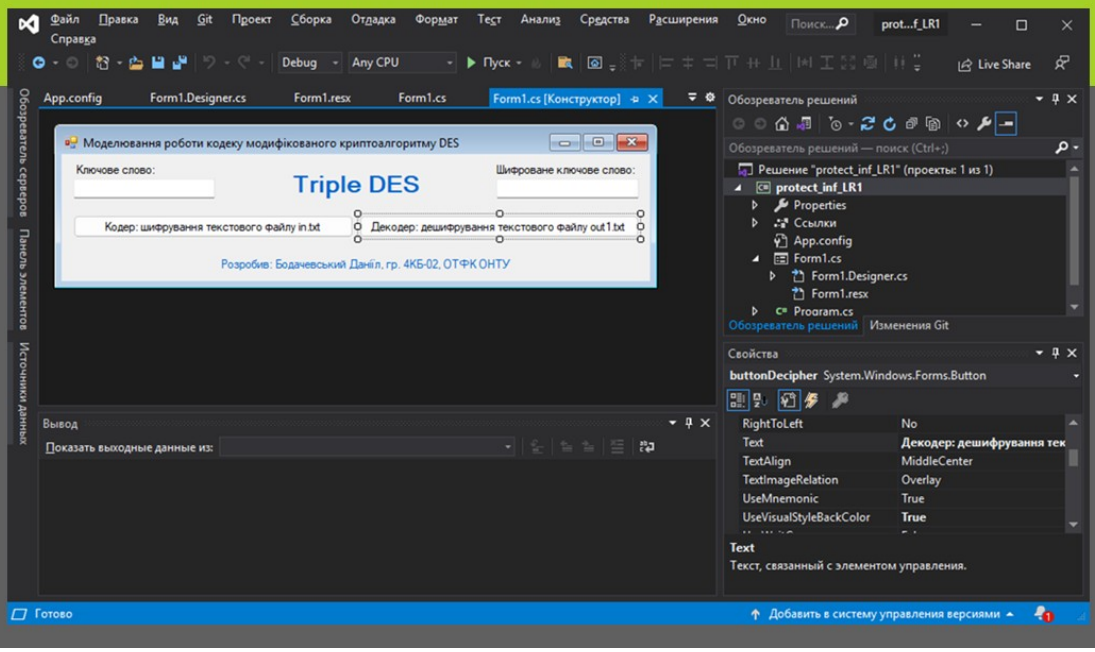


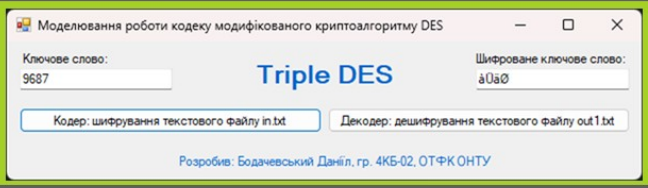
Блок-схема алгоритму дій користувача

UML-діаграма введення та підготовки даних для обробки

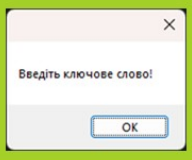


Інтерфейс Microsoft Visual Studio 2022 C# у режимі Windows Forms

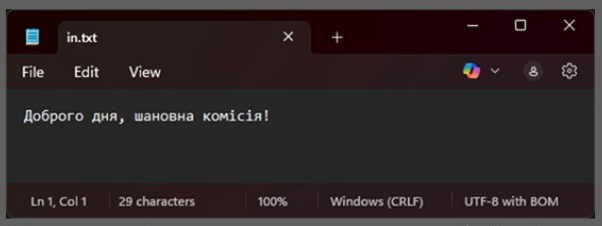




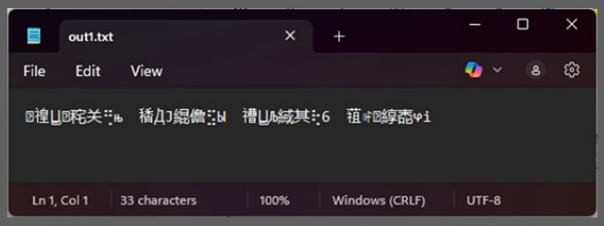
Інтерфейс програми-кодеку на базі алгоритму Triple DES



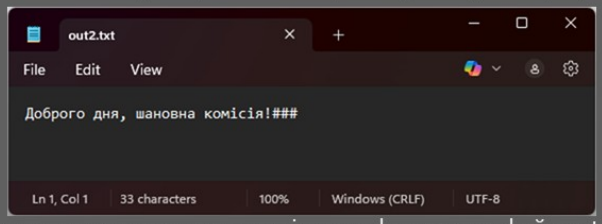
Виведення повідомлення у разі відсутності ключового слова у полі



Приклад вмісту початкового текстового файлу in.txt



Приклад виведення результатів шифрування у файл out1.txt



Приклад виведення результатів дешифрування у файл out2.txt

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Бодачевського Данііла Васильовича

(прізвище, ім`я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Безпека комп'ютерних систем і мереж»*

Керівник дипломного проекту (роботи) *Кривченко Юрій Вікторович*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Моделювання роботи кодеку модифікованого криптоалгоритму DES*

Обсяг розрахунково-пояснювальної записки *76* сторінок

Обсяг графічної (презентаційної) частини *16* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

Представлений дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений створенню програмної моделі кодеку модифікованого криптоалгоритму DES та її реалізації мовою програмування C# і складається з пояснювальної записки та мультимедійної презентації з відповідними схемами.

б) характеристика виконання кожного розділу дипломного проекту

Пояснювальна записка складається з основного розділу (Аналітичний огляд методів і засобів симетричної криптографії; Визначення структури і властивостей криптоалгоритму DES; Аналіз існуючих модифікацій криптоалгоритму DES; Моделювання роботи алгоритму модифікованого DES у CrypTool2; Підготовка засобів розробки; Реалізація програмного застосунку), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту

Графічна частина складається з 16 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять структурні, та функціональні схеми, діаграми, блок-схеми алгоритмів, скріншоти передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання пояснювальної записки відмінна, розробку виконано у повному обсязі.

г) перелік позитивних якостей дипломного проекту *Автор обґрунтовує необхідність вдосконалення класичних алгоритмів через їх обмеження, що додає практичної та теоретичної цінності дослідженню. Робота містить схеми, таблиці, UML-діаграми і послідовні пояснення кожного етапу алгоритму. Використання CrypTool2, практична реалізація з використанням C# дозволяють продемонструвати спроможність системи до криптографічного захисту*

д) основні недоліки дипломного проекту *Для забезпечення вищої криптостійкості варто було розглянути також інші режими роботи кодеку, наприклад CBC.*
Деякі розділи, особливо пояснення базових принципів DES і хід моделювання, містять великі обсяги описів матеріалу.

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові рецензента *к.т.н. Рудніченко Микола Дмитрович*

Місце роботи і посада рецензента *Національний університет «Одеська політехніка», доцент кафедри інформаційних технологій*

Підпис

« 20 » червня 2025 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Бодачевського Данiїла Васильовича

(прізвище, ім`я та по батькові)

Спеціальність: *123 «Комп'ютерна інженерія»*

Освітня програма: *«Безпека комп'ютерних систем і мереж»*

Тема дипломного проекту: *Моделювання роботи кодеку модифікованого криптоалгоритму DES*

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка до дипломного проекту містить 76 сторінок. У пояснювальній записці описано створення програмної моделі кодеку модифікованого криптоалгоритму DES та її реалізації мовою програмування C#. Графічна частина складається з 16 слайдів, оформлених у вигляді презентації, передбачених технічним завданням. Якість виконання пояснювальної записки та слайдів добра.*

б) самостійність роботи над проектом: *Протягом виконання дипломного проекту здобувач освіти Бодачевський Данiїл поступово та послідовно виконував всі етапи, проявив ініціативу в створенні загальної концепції та реалізації роботи. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника.*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Бодачевський Данiїл під час роботи над дипломним проектом вивчив достатньо багато літературних та інтернет-джерел за даною тематикою.*

Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту проекту.

г) вміння розв'язувати виробничі та конструкторські питання *Під час виконання дипломного проекту здобувач освіти Бодачевський Даніїл показав вміння організовано працювати над поставленим завданням, застосовувати знання у сфері безпеки комп'ютерних систем і мереж, програмування, використовуючи сучасні комп'ютерні програмні засоби розробки, такі як Microsoft Visual Studio 2022, CrypTool2.*

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові керівника дипломного проекту *Кривченко Юрій Вікторович*

Місце роботи і посада керівника дипломного проекту *ВСП «Одеський технічний фаховий коледж ОНТУ», викладач спецдисциплін циклової комісії комп'ютерних технологій та програмної інженерії, голова ЦК*

Підпис

«14» червня 2025 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Бодачевський Д.В.,
здобувач освіти гр. 4КБ-02, та

Кривченко Ю.В.,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Моделювання роботи кодеку модифікованого криптоалгоритму DES» (автор роботи – Бодачевський Д.В., керівник роботи – Кривченко Ю.В.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Бодачевський Д.В. /

Керівник _____ / Кривченко Ю.В. /

«16» червня 2025 р.

ДОВІДКА

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти IV курсу
відділення комп'ютерних систем групи 4КБ-02

Бодачевського Даніїла Васильовича

на тему *Моделювання роботи кодека модифікованого криптоалгоритму DES*

Висновок відповідальної особи за проведення нормоконтролю:
пояснювальна записка до дипломного проєкту виконана з несуттєвими порушеннями ДСТУ та оформлена відповідно до вимог Положення про дипломне проєктування

(підпис) *16.06.2025* (дата) *Петрашова В.І.* (П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного плагіату *згідно звіту про перевірку від 29.05.2025 р. значення коефіцієнту подібності в роботі становить 25,69%, коефіцієнт цитування – 2,55%.*

(підпис) *16.06.2025* (дата) *Краснокутська К.Г.* (П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти *Бодачевського Д.В.* (П.І.Б.)

проведена « *16* » *червня* 2025 р.

Висновки *Пояснювальна записка до дипломного проєкту виконана у повному обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає вимогам Положення про дипломне проєктування та рекомендована до захисту.*

Голова ЦК КТ та ПІ (підпис) *Кривченко Ю.В.* (П.І.Б.)

StrikePlagiarism.com

Дата звіту 5/29/2025
Дата редагування 5/29/2025

Документ прийнятий

Звіт подібності

метадані

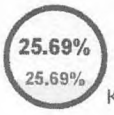
Назва організації
Odesa Technical Professional College of Odesa National University of Technology
Заголовок
Моделювання роботи кодеку модифікованого криптоалгоритму DES

Автор
Бодачевський Даніїл Васильович
Науковий керівник / Експерт
Кривченко Юрій Вікторович

підрозділ
Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

КЦ

25
Довжина фрази для коефіцієнта подібності 2

14288
Кількість слів

112782
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		22
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки		0
Парафрази (SmartMarks)	a	128

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	http://openarchive.nure.ua/bitstream/document/11051/1/2019_M_KRiSTZI_Kulish_D_A.docx	559 3.91 %
2	Розробка програмної моделі генерування та валідації надійних паролів [redacted] 5/27/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	406 2.84 %
3	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	103 0.72 %

4	Розробка програмної моделі генерування та валідації надійних паролів [redacted] 5/27/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	94 0.66 %
5	https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download	75 0.52 %
6	Розробка програмної моделі генерування та валідації надійних паролів [redacted] 5/27/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	75 0.52 %
7	http://openarchive.nure.ua/bitstream/document/11051/1/2019_M_KRiSTZI_Kulish_D_A.docx	64 0.45 %
8	https://card-file.ontu.edu.ua/bitstreams/5240e379-7721-49f0-8ee8-27140b0b473a/download	60 0.42 %
9	https://academic.csuohio.edu/yuc/security/Chapter_06_Data_Encription_Standard.pdf	59 0.41 %
10	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	58 0.41 %

з домашньої бази даних (9.64 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Розробка програмної моделі генерування та валідації надійних паролів [redacted] 5/27/2025 **Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")**	**1313 (76) 9.19 %**
2	Розробка моделі оптимізації та безпеки передачі даних за допомогою механізму NBAR [redacted] 5/28/2025 **Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")**	**64 (3) 0.45 %**

з програми обміну базами даних (0.05 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Кладько БК-812м.docx [redacted] 12/18/2023 National University "Zaporizhzhia Polytechnic" (Кафедра "Інформаційна безпека та наноелектроніка")	7 (1) 0.05 %

з Інтернету (16.00 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	**http://openarchive.nure.ua/bitstream/document/11051/1/2019_M_KRiSTZI_Kulish_D_A.docx**	**849 (18) 5.94 %**
2	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content**	**326 (8) 2.28 %**
3	**https://openarchive.nure.ua/server/api/core/bitstreams/2214ba81-f148-4fcf-87d6-f764230b3617/content**	**168 (7) 1.18 %**
4	**https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download**	**130 (5) 0.91 %**
5	**https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download**	**73 (4) 0.51 %**
6	**https://card-file.ontu.edu.ua/bitstreams/5240e379-7721-49f0-8ee8-27140b0b473a/download**	**72 (2) 0.50 %**

7	https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download	72 (6) 0.50 %
8	**https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download**	**65 (3) 0.45 %**
9	**https://academic.csuohio.edu/yuc/security/Chapter_06_Data_Encription_Standard.pdf**	**59 (1) 0.41 %**
10	**https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download**	**57 (2) 0.40 %**
11	**https://card-file.ontu.edu.ua/bitstreams/549ee9fe-7574-4ae5-b500-9fe2711f33e6/download**	**54 (3) 0.38 %**
12	**https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download**	**46 (2) 0.32 %**
13	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content	44 (4) 0.31 %
14	**http://cpsm.kpi.ua/stud/bak/DP_BAK_KARAULOVA_LU.pdf**	**32 (1) 0.22 %**
15	https://card-file.ontu.edu.ua/server/api/core/bitstreams/995bdcec-4e4d-4321-8070-4d6badcb8e49/content	30 (2) 0.21 %
16	https://card-file.ontu.edu.ua/server/api/core/bitstreams/4bb7255e-46d4-4349-9726-9698476da02d/content	27 (4) 0.19 %
17	https://card-file.ontu.edu.ua/bitstreams/0e72a3b9-bdd7-4711-a3c6-dedc1d4287cc/download	27 (3) 0.19 %
18	https://card-file.ontu.edu.ua/bitstreams/c58b0ff5-46e0-49f8-8cbe-65c32256665d/download	26 (2) 0.18 %
19	https://card-file.ontu.edu.ua/bitstreams/bbaf3f38-16a8-4070-bead-5562769b7c71/download	23 (1) 0.16 %
20	https://card-file.ontu.edu.ua/server/api/core/bitstreams/f5042058-9544-4ac7-bd33-42bd733c8e6b/content	19 (2) 0.13 %
21	http://opcb.kpi.ua/wp-content/uploads/2021/08/Lekc_4_BGD_CZ_2021.pdf	18 (1) 0.13 %
22	https://card-file.ontu.edu.ua/bitstreams/e4afae26-0a7e-4a4d-afc2-94341838de2a/download	17 (2) 0.12 %
23	https://card-file.ontu.edu.ua/bitstreams/21173711-5b67-4b87-b17f-6302c25e7a31/download	16 (2) 0.11 %
24	https://forums.codeguru.com/showthread.php?476466-3DES-(Triple-DES)-in-C	14 (1) 0.10 %
25	https://pastebin.com/7zHbE3pe	13 (1) 0.09 %
26	https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download	9 (1) 0.06 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»_
Освітньо-професійна програма: **«Безпека комп'ютерних систем і мереж» Група:** 4КБ- 02

Дипломний **проект здобувача освіти денної форми навчання КБ.** 02.04.000.ДП

БОДАЧЕВСЬКОГО
ДАНІЇЛА ВАСИЛЬОВИЧА

м. Одеса
2025 **р.**

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»_