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-01

Дипломний проект

здобувача освіти денної форми навчання
КБ.01.08.000.ДП

КЕКУЛ
ДМИТРА КИРИЛОВИЧА

м. Одеса
2024 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-01**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Програмна реалізація стегосистеми для передачі і захисту прихованих даних

Проектний матеріал складається з пояснювальної записки на 85 сторінках та графічного (презентаційного) матеріалу на 16 аркушах (слайдах)

Дипломник _____ (Кекул Д.К.)

Керівник _____ (Скорняков В.С.)

Консультанти:

з економічного розділу _____ (Іванченков В.С.)

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 _____ (Скорнякова О.В.)

Захист «19» 06 2024 р. Протокол ЕК № 3

Оцінка ЕК 5 (відмінно) 95 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.*

"15" 04 2024 р.

ЗАВДАННЯ

на дипломний проект

Здобувачеві освіти *Кекул Дмитра Кириловича*

(прізвище, ім'я, по батькові)

1. Тема проекту *Програмна реалізація стегосистеми для передачі і захисту прихованих даних*

затверджена наказом по коледжу від "2" 11 2023 р. № 244-А2-ОД

2. Термін здачі закінченого проекту 10.06.2024

3. Вихідні данні до проекту *1. Реалізувати програмну модель стеганографічної системи для приховування файлів різного типу з шифруванням AES; 2. Забезпечити використання у якості контейнерів графічних та звукових файлів; 3. Реалізувати візуальний інтерфейс користувача для створюваного додатку; 4. Використовувати мову C# для реалізації програмного застосунку стегосистеми; 5. Забезпечити перевірку наявності прихованої інформації у контейнері*

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

Складання моделі стеганографічної системи; Аналіз існуючих програмних рішень для стегосистем; Аналіз структури стеганографічної системи; Застосування технологій стеганографії та шифрування у проекті; Розробка структури стегосистеми для передачі і захисту прихованих даних; Розробка структури діаграми класів стегосистеми для передачі і захисту прихованих даних; Розробка візуального інтерфейсу застосунку стегосистеми

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

Узагальнена модель стеганографічної системи; Стеганографічний алгоритм приховування повідомлення; Принцип приховування інформації у зображенні за методом LSB; Реалізація шифрування за алгоритмом AES; Застосування дерева Хаффмана; Загальна структура стегосистеми; Структура формату зберігання у контейнерах секретних файлів; Діаграма взаємодії класів у програмній реалізації стегосистеми; Головне та дочірнє вікно програмного застосунку стегосистеми приховування та відновлення інформації

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

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

7. Дата видачі завдання 15.01.2024

Керівник *Скорняков В.С.* _____
(підпис)

Завдання прийняв до виконання *Кекул Д.К.* _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	*Вступ. Постановка задачі проектування*	29.04.24	виконано
2	*Аналіз технічного завдання та пошук літератури*	30.04.24	виконано
3	*Складання моделі стеганографічної системи*	02.05.24	виконано
4	*Аналіз існуючих програмних рішень для стегосистем;*	05.05.24	виконано
5	*Аналіз структури стеганографічної системи;*	07.05.24	виконано
6	*Застосування технологій стеганографії та шифрування*	10.05.24	виконано
7	*Розробка структури стегосистеми*	15.05.24	виконано
8	*Розробка структури діаграми класів стегосистеми*	21.05.24	виконано
9	*Розробка візуального інтерфейсу застосунку*	28.05.24	виконано
	стегосистеми		
10	*Реалізація програмного застосунку стегосистеми*	01.06.24	виконано
	для приховування користувацьких файлів		
11	*Випробування застосунку та аналіз результатів*	03.06.24	виконано
12	*Виконання економічних розрахунків*	04.06.24	виконано
13	*Розробка питань з охорони праці та техніки безпеки*	05.06.24	виконано
14	*Підготовка мультимедійної презентації проекту*	09.06.24	виконано

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КБ 01. 08 000. 00 ДП	Дипломний проект		
А4			КБ 01. 08 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КБ 01. 08 000. 00 ДП		
Розробив		Кекул Д.К.		10.06	Програмна реалізація стегосистеми для передачі і захисту прихованих даних	Літ.	Аркуш 4 / Аркушів 85
Перевірив		Скорняков В.С.		10.06		Н Д П	
Н. контр		Петрашова В.І.		10.06	ВСП "ОТФК ОНТУ" гр.4КБ-01		
Затверд.		Кривченко Ю.В.		10.06			

ЗМІСТ

ВСТУП

Цифрова стегано-графія приховує сам факт передавання чи зберігання даних, що досягається шляхом впровадження даних, що захищається, у різні мультимедійні об'єкти (контейнери), котрі не втрачають з цього своїх споживчих властивостей. В комп'ютерній стеографії задля цього використовуються файли різних форматів, мережеві пакети та т.д. Із іншого боку, приховання інформації треба застосовувати у не комерційному секторі, аби сховати дані, яку хтось хоче зберігати у секреті.

Стегано-графія стала доступна задля більшості користувачів та спроможне застосовуватися у протизаконних цілях, наприклад, задля несанкціонованої передавання комерційних чи державних секретів; переписки терористичних угруповань. Тому із'являється необхідність в розробці ефективних методів виявлення схованих вкладень, у мультимедійних об'єктах, переданих у комп'ютерних мережах. Комп'ютерні технології надали нового імпульсу розвитку стеографії, із'явилася комп'ютерна стегано-графія, що забезпечила непомітне, із позицій споживчих якостей, вбудовування інформації у файли-контейнери, що містять у цифровому вигляді звуку чи фото. Інтерес щодо цієї області залишається на високому рівні, хоча вже існує багато застосувань стеографії на практиці. Прикладами таких застосувань є:

– захист даних з несанкціонованого доступу;

– протидія системам моніторингу і керування ресурсами мереж;

– маскування програмного забезпечення з незареєстрованих користувачів;

– захист авторського права на деякі види інтелектуальної власності.

Метою даного дипломного проекту є програмна реалізація стего-системи задля передавання та захищення схованих інформації, що вміщує надати зручні і надійні засоби задля приховання інформації в різних форматах файлів. Розроблювана стегосистема вміщує ефективно приховувати сповіщення всередині медіа-файлів, забезпечуючи у цьому стійкість щодо аналізу і виявлення.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Складання моделі стеганографічної системи

Основною задачею стеганографічної системи є розміщення вхідного сповіщення у контейнері цим способом, аби будь-що стороння людина не змогла помітити нічого, крім його основною вмісту. Основний вміст бокса не відіграє ніякої ролі ні задля відправника, ні задля одержувача, котрих цікавить лише успішна передача сповіщення, вміщеного у ньому (стеганограми). Потрібно обов'язково враховувати те, що сам факт відправлення бокса з автора щодо одержувача не повинен виглядати дивним, але разом з цим не повинно спостерігатись помітних відхилень бокса з норми.

Узагальнена модель стеганографічної системи схематично представлена на рис. 1.1.

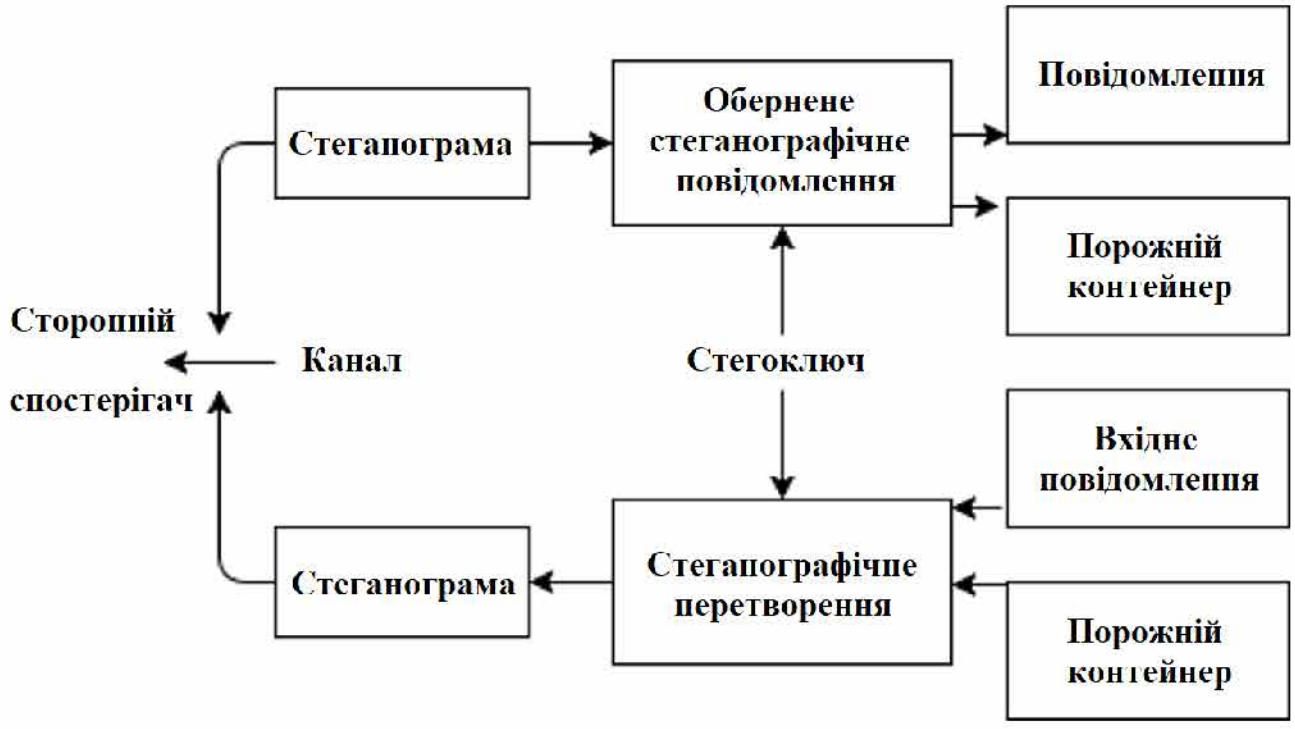


Рисунок 1.1. Узагальнена модель стеганографічної системи

1.2 Визначення характеристик стеганографічної системи

Вкраплення сповіщення у бокс цим способом, аби будь-котрий сторонній спостерігач не зміг помітити різниці поміж оригінальним контейнером і модифікованим, є задачею будь-якої стеганографічної системи. Зазвичай система

будується так аби забезпечити певний компроміс її базових характеристик, щодо котрих відносяться невідчутність, стійкість, безпека, пропускна здатність створюваного стеганоканалу і обчислювальна складність реалізації.

Невідчутність. Вкраплення сповіщення повинне зберігати перцепційну якість оригінального бокса. Задля аудіосигналів сповіщення повинне бути невідчутним, задля фото – візуально непомітним. Невідчутності сповіщення треба досягнути внесенням мінімальних модифікацій у стеганоперетворенні бокса, наприклад, на рівні похибки дискретизації у оцифровці. Крім того, досягти невідчутності допомагає врахування властивостей систем людського слуху і зору.

Чисельними показниками невідчутності на практиці часто стають співвідношення сигнал/шум SNR, максимальна різниця MD, середньоквадратична похибка MSE і інші.

Стійкість. Суть поняття стійкості залежить з типу атак, котрі характерні задля тієї чи іншої стеганографічної системи. Так, задля систем схованої передавання інформації найбільш характерними є пасивні атаки, тому в цьому разі під стійкою насамперед розуміють систему, що здатна ефективно їм протидіяти.

Стійкість задля інших видів стеганосистем, мов правило, оцінюють через число помилок, що виникли у вилучені сповіщення із бокса легальним користувачем опісля можливих спотворень цього бокса ненавмисними чи активними атаками. Наприклад, дослідження стійкості щодо процесів друку і сканування, що обов'язково супроводжують стеганоконтейнер в задачах захищення даних на паперових носіях.

Застосуванням системи визначається необхідний рівень стійкості. Так, говорячи про стійкість щодо активних атак, виділяють системи ЦВЗ зі стійкими, крихкими і напівкрихкими водяними знаками.

Розглянемо детальніше стійкість в моделях пасивного і активного противників. Стеганосистема і відповідно стеганоконтейнери, котрі вона продукує, вважаються стійкими щодо пасивних атак тоді та тільки тоді, коли

несанкціонований користувач не вміщує можливості відрізнити пусті контейнери з заповнених, зокрема методами візуального і статистичного аналізу.

Велика частина поширених програмних продуктів задля схованої передавання даних методами комп'ютерної стегографії, реалізують різні модифікації способу найменшого значущого біту, суть якого полягає в заміні молодших біт бокса бітами приховуваного сповіщення. Користувач обирає довільний бокс, розміри якого дозволяють розмістити в ньому сповіщення, та у результаті отримує стеганоконтейнер, що візуально не відрізняється з пустого. Поміж молодшими бітами сусідніх складових природних контейнерів, але разом з цим поміж молодшим і іншими бітами складових бокса існує кореляційний зв'язок, що спроможне бути порушеним вкрапленням сповіщення. В цьому разі задля виявлення стеганоконтейнеру достатньо найпростішого аналізу – візуального аналізу бітових зрізів. Мов правило, через наявність похибки дискретизації у оцифровці і інших шумів цифрові контейнери, що отримані із аналогових, більш стійкі щодо такої атаки, ніж ті, що були створені відразу цифровими. Разом із тим, вкраплюючи сповіщення у НЗБ зашумленого бокса, треба розподіляти його по всьому об'єму молодших біт, інакше різниця поміж не зміненою і зміненою вкрапленням частинами спроможне бути виявлена візуальною атакою на відповідний бітовий зріз.

Ємність. Ємність визначається мов максимальна число інформації сповіщення, котрі можуть бути вкрапленими у один елемент бокса із дотриманням вимог невідчутності і стійкості.

Існують різні, іноді діаметрально протилежні підходи щодо визначення кількості приховуваної даних на сьогоднішній день. Ці розбіжності обумовлені відмінностями у цілях захищення даних, видах порушника, їх можливостях, типах контейнерів і повідомлень і іншими факторами. Зокрема, у якості теоретично досяжних границь, що не залежать з особливостей практичного впровадження, використовують оцінку пропускної здатності, отриману у теоретико-інформаційній моделі стеганосистеми.

Ємність визначає потенційний об'єм даних, яку треба сховати тим чи

іншим методом стегографії. АЛЕ той об'єм, що був реально використаний у процесі стеганоперетворення певного бокса (тобто вкраплення у нього додаткової даних), будемо називати наповненістю бокса. Очевидно, що у рамках тієї чи іншої стеганосистеми наповненість будь-якого бокса не спроможне перевищувати пропускної здатності створюваного нею стеганоканалу. Наповненість зручно вимірювати в відсотках з пропускної здатності. Так, наповненість заповненого бокса складає 100%, порожнього – О%.

1.3 Визначення способів приховання інформації

Багато змін трапилося із вітчизняними носіями завдяки застосуванню стегографії у комп'ютерних технологіях. Ці носії можуть бути віднесені щодо багатьох видів інформації, таких мов текст, диск, аудіо, фото, звук, мережевий трафік чи інші дані цифрової передавання інформації. Способи приховання інформації наведено нижче.

Приховання у тексті. Задля приховання даних в тексті (лінгвістична стегано-графія) застосовується звичайна надлишковість письмової мови чи формати представлення тексту.

Найскладнішим об'єктом задля приховання є електронна версія тексту, тому що його друкована версія спроможне бути зображенням у електронному вигляді, обробленим відповідними методами. Ця складність у основному обумовлена відносним дефіцитом в тексті надлишковості, на відміну з фото чи аудіо- файлу. У той час мов існує можливість внести невидимі задля ока модифікації в фото чи не відчутні задля слухової системи людини (ССЛ) зміни в звучанні аудіофайлу, будь-що зайва літера, зайвий символ чи зайвий знак пунктуації спроможне бути виявлений випадковим читачем.

Існують три основні моделі приховання інформації в тексті, що найширше розповсюджені:

- моделі довільних інтервалів;
- синтаксичні моделі;
- семантичні моделі.

Приховання у зображеннях. У більшості випадків використовуються стеганографічні моделі із графічними зображеннями у ролі контейнерів саме через ці причини:

- розповсюдження цифрових фотографій і відео, котрі треба захищати з протизаконного тиражування і розповсюдження;
- відносно великий об'єм графічних фото, що дає широкий простір задля приховання інформації (великого розміру);
- розмірність бокса відомий заздалегідь, що дає змогу обирати оптимальний бокс;
- відносно слабка чутливість людського ока щодо незначних змін в цифрових графічних зображеннях;
- добре розроблені, у останній час, моделі цифрової обробки фото.

Приховання в відео-файлах. Стеганографічні моделі приховання рідше поза все використовуються в відеоданих, оскільки даний файл складається із динамічних фото (фреймів) і звукової доріжки. Варто разом з цим зазначити, що досі не використовуються мов контейнери одночасно аудіодоріжки і фрейми.

На сьогодні існує три моделі задля приховання даних в відеоданих, але саме:

Метод вбудовування на рівні масштабів – біти приховуваного сповіщення вбудовуються у коефіцієнти ДКП. Враховуючи, що використовуються алгоритми стиснення, основною проблемою стає накопичення зсуву і помилок. Задля зменшення внесених змін використовують додатковий спеціальний сигнал. У зв'язку із обмеженням бітової швидкості у вбудовуванні змінюється лише 10-12% масштабів ДКП. У використанні даного способу приховувана інформація зберігається у фільтруванні, зашумленні (адитивним шумом) та дискретизації.

Метод вбудовування даних на рівні бітової площини – відрізняється високою пропускною здатністю та легкими обчисленнями. Але є й істотний недолік: інформація, вбудована цим способом, спроможне бути легко видалена. У повторному накладенні послідовності біт якість відео погіршиться, але приховувана інформація буде знищена.

Метод вбудовування даних поза рахунок енергетичної різниці поміж коефіцієнтами – у основі лежить диференціальне вбудовування енергії. Цей метод спроможне використовуватись задля багатьох алгоритмів стиснення. Інформація вбудовується шляхом видалення декількох масштабів ДКП.

У основному, приховання у відео використовує моделі, котрі використовуються задля приховання звуку і фото, оскільки вже є відеозображеннями фото та звуків. Відео складається із переміщення фото у супроводі із аудіо. Насправді це є перевагою, оскільки будь-котрі невеликі спотворення користувачі навіть не помітять через неперервну число інформації.

Приховання у аудіо-файлах. Особливий розвиток отримали стеганографічні моделі приховання даних в аудіосередовищі. Це охарактеризовано тим, що ССЛ працює в надширокому динамічному діапазоні та вміщує доволі малий різницевий діапазон. Виходячи із цього, треба зробити висновок, що в аудіофайлах присутній широкий простір задля приховання інформації. Разом з цим ССЛ не здатна розрізняти абсолютну фазу, вирізняє лише відносну. Крім того, існують деякі види спотворень, викликаних зовнішнім середовищем, котрі треба використати задля приховання інформації. Приховання інформації у аудіо файлах особливо складне через його великий діапазон частот. Аудіосигнали разом з цим чутливі щодо випадкових шумів. Шум спроможне бути виявлений, коли він знаходиться у діапазоні з одного щодо мільйону в звукових файлах. У приховуванні звуку користувач повинен скористатися перевагами слабкості людського слухового апарату, але разом з цим слід подбати про його високу чутливість.

1.4 Аналіз існуючих програмних рішень задля стегосистем

В даному підрозділі буде виконано короткий аналіз сучасних програмних засобів стегографії, визначено їх переваги і недоліки.

1.4.1 Програма ImageSpyer G2 задля приховання даних

ImageSpyer G2 є утилітою задля приховання даних в графічних файлах із використанням зашифровування (рис.1.3).

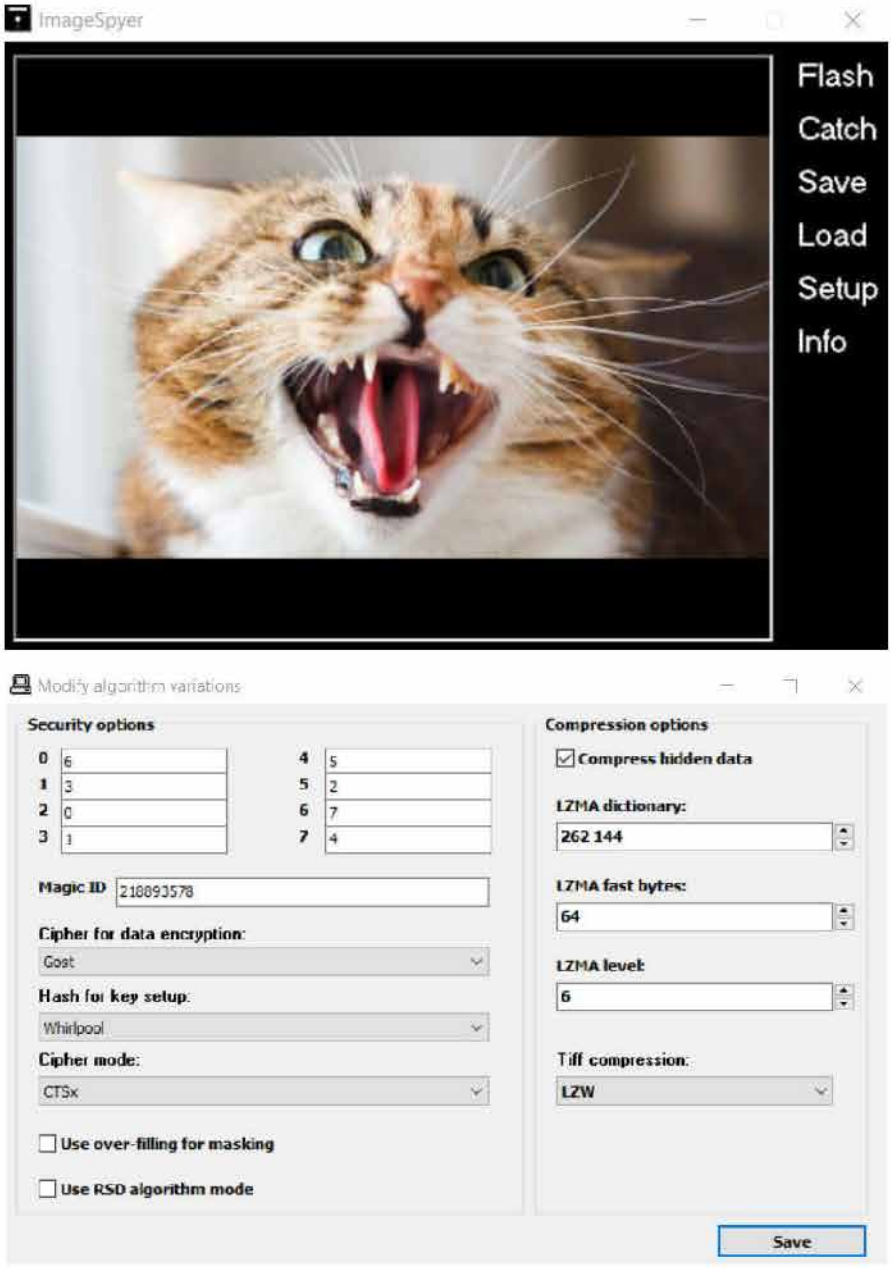


Рисунок 1.3. Інтерфейс утиліти ImageSpyer G2

У цьому підтримується близько 30 алгоритмів зашифровування і 25 хеш-функцій задля зашифровування бокса. Приховує обсяг, що дорівнює числу пікселів фото. Опціонально доступна компресія інформації, що приховуються. Програма вміщує ці властивості:

– підтримує близько 30 алгоритмів зашифровування і 25 хеш-функцій задля зашифровування бокса;

– можливість сховати обсяг, що дорівнює числу пікселів фото;

– опціональна компресія інформації, що приховуються;

– не згадується про типи графічних файлів, котрі підтримуються;

– не вказано, чи можливо видобування схованих інформації безпосередньо із програми.

1.4.2 Програма RedJPEG задля приховання даних

Проста в використанні програма RedJPEG призначена задля приховання будь-котрих інформації в JPG-зображенні поза поміччю стеганографічного способу (рис.1.4).

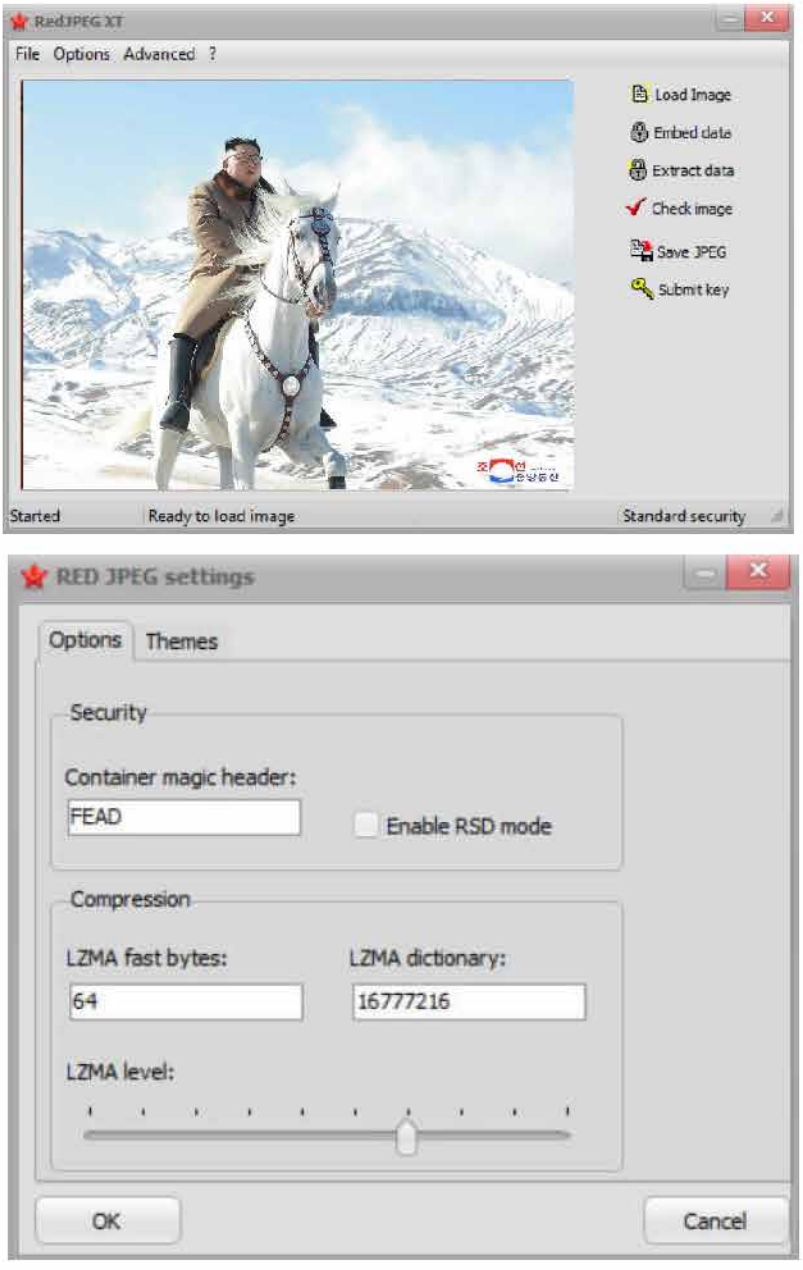


Рисунок 1.4. Інтерфейс утиліти RedJPEG

Програма RedJPEG використовує відкриті алгоритми зашифровування, потоковий шифр AMPRNG і Cartman II DDP4 в режимі хеш-функції, LZMA-

компресію. Програма вміщує ці властивості:

– простий в використанні інтерфейс;

– приховання інформації в форматі JPG поза поміччю стеганографічного способу;

– впровадження відкритих алгоритмів зашифровування і компресію LZMA;

– не згадується про підтримку інших графічних форматів;

– не вказано, чи можливе видобування схованих інформації безпосередньо із програми.

1.4.3 Програма Hide'N'Send задля приховання даних

Програма Hide'N'Send є одним із потужних інструментів стегографії фото. Він включає зашифровування і приховання інформації в файлі фото (формат JPG), разом з цим шифрує дані поза поміччю алгоритму стегографії F5. Приховання інформації здійснюється поза поміччю алгоритму ISB (найменшого результати) задля стегографії фото. Замість того, аби ховатися у структурі файлу, ці алгоритми приховують дані всередині фото. Інтерфейс інструмента простий та пропонує дві вкладки (рис. 1.5) − одна задля приховання інформації, але інша − задля вилучення інформації.

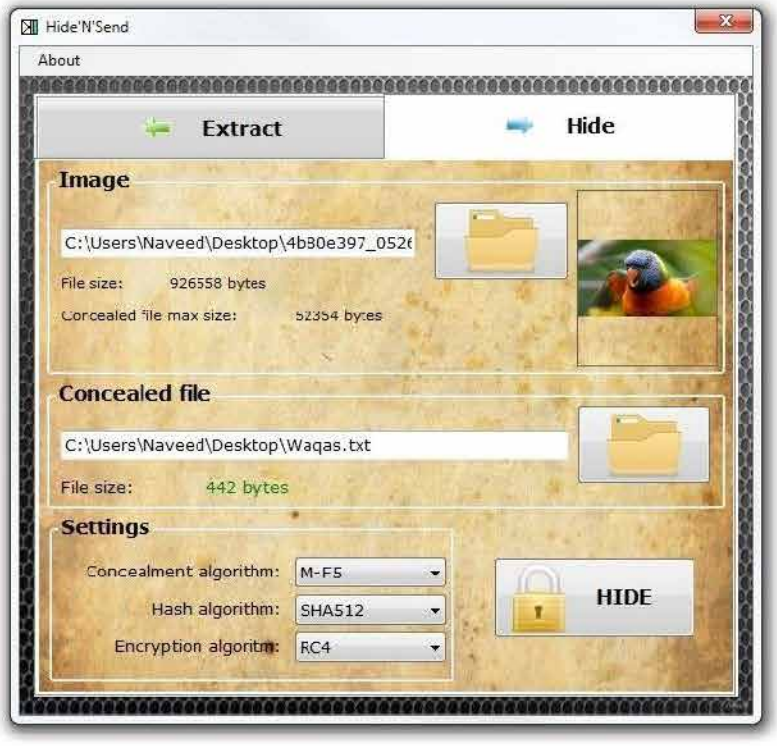


Рисунок 1.5. Інтерфейс утиліти Hide'N'Send

У використанні утиліти Hide'N'Send є можливість вибрати відповідні параметри. Треба просто запустити інструмент, обрати файл фото, але потім обрати файл, котрий потрібно сховати, визначивши тип зашифровування, та сховати дані у зображенні. Задля того, аби витягти приховану дані на зображенні, потрібно застосовувати той самий інструмент.

1.4.4 Програма SteganPEG задля приховання даних

Програма SteganPEG використовує формат фото JPG задля реалізація стегографії (рис. 1.6). Програма спроможне сховати будь-котрі приватні дані чи секретні сповіщення у зображенні JPG, не змінюючи якість фото. У результаті неможна візуально знайти різницю поміж стеганографічним зображенням та звичайним зображенням, коли застосовувати формат JPG. Інтерфейс програмного забезпечення простий, із мінімальною кількістю опцій.

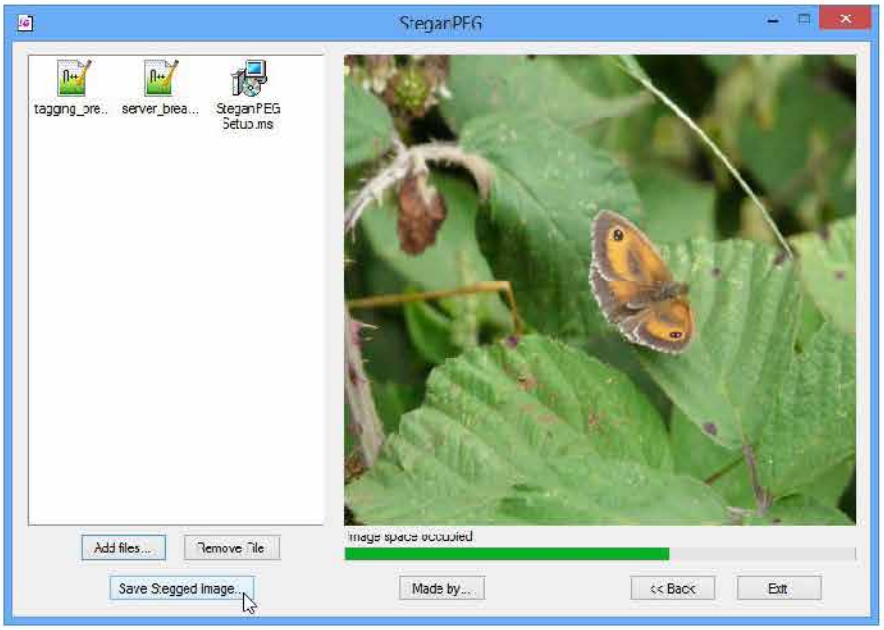


Рисунок 1.6. Інтерфейс утиліти SteganPEG

1.4.5 Програма OpenStego задля приховання даних

Програма OpenStego забезпечує приховання інформації, але разом з цим створення водяних знаків. Використовуючи OpenStego, треба ефективно виконувати стеганографію із файлами фото типу JPG, JPG, BMP, GIF, PNG тощо. Результатом OpenStego є файл PNG. OpenStego є безкоштовним інструментом із відкритим вихідним кодом, розробленим поза поміччю мови Java. застосовується

задля виявлення несанкціонованих копій файлів фото завдяки водяним знакам (рис. 1.7).

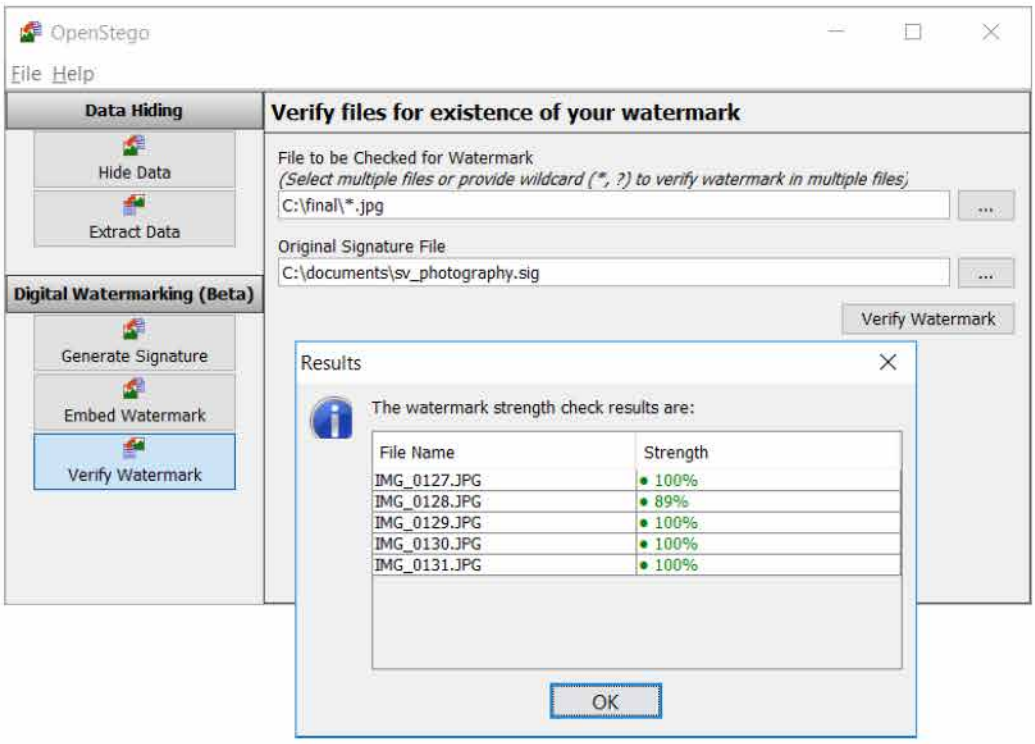


Рисунок 1.7. Інтерфейс утиліти OpenStego

1.4.6 Програма Our Secret задля приховання даних

Програмний інструмент Our Secret (рис. 1.8) – це ще один інструмент стегографії фото, котрий надає сховати файли, текст, сповіщення на фотографіях.

Рисунок 1.8. Інтерфейс утиліти Our Secret

Вхідним файлом бажано є файл JPG із збереженням невеликого розміру

фото. Поза поміччю утиліти Our Secret треба переносити разом з цим аудіофайли в файлах фото.

1.4.7 Програма Xiao Steganography задля приховання даних

Програма Xiao Steganography – це безкоштовний інструмент стегографії, котрий треба застосовувати задля приховання секретних файлів в зображенні, але разом з цим аудіофайлів (рис. 1.9). Найбільш часто використовувані формати файлів – це BMP задля фото та WAV задля аудіофайлів. Задля роботи треба відкрити цей інструмент, завантажити у нього потрібні файли і секретне сповіщення. У програмі треба вибрати будь-котрий із алгоритмів зашифровування DES, DES112, RC2. Хешування включає SHA, MD4, MD2 та MD5. Неможна розпакувати прихований файл поза поміччю будь-якого іншого програмного забезпечення.

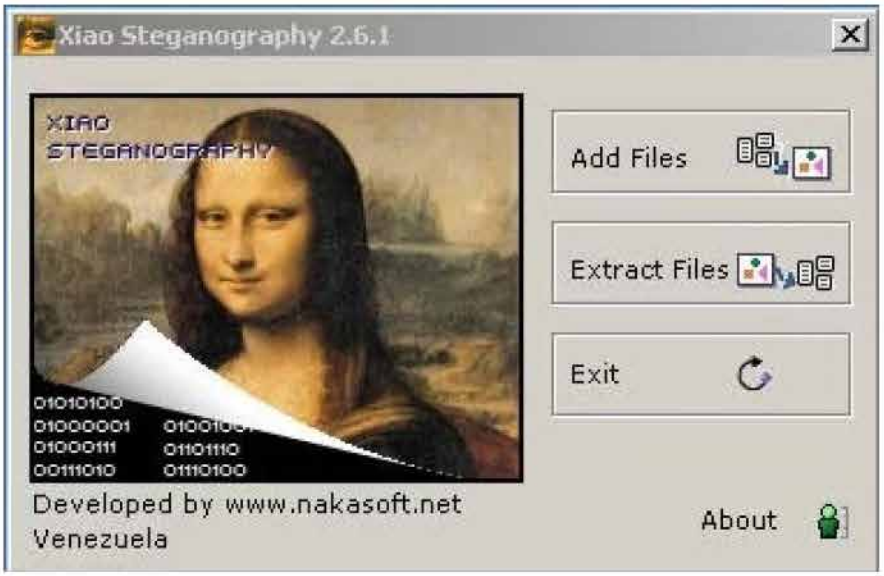


Рисунок 1.9. Інтерфейс утиліти Xiao Steganography

1.5 Аналіз вимог щодо стего-системи

Розглянуті в підрозділі 1.4 існуючі інструменти задля приховання інформації в файлах дозволяють отримати бажані результати в стегографії. Але кожна із розглянути утиліт передбачає впровадження свого формату та алгоритму, отже всі ці утиліти несумісні поміж собою у шифруванні-дешифруванні. Стегано-графія була розроблена задля безпечного спілкування. Однак злочинці і терористичні організації разом з цим почали застосовувати

подібні інструменти. Немає простого способу виявити прихований файл в файлі, неможна у відкритті файлу побачити, чи є там прихована інформація, вміщує бути належне спостереження.

Стегано-графія є ефективним способом захищення даних, котрий стає особливо актуальним, але деякі існуючі програмні реалізації є складними системами чи несумісними поміж собою, обмеженим по функціональності. ІЗ цієї причини виникла потреба в розробці програмного застосунку задля передавання та захищення схованих інформації. Треба організувати застосунок цим способом, аби задля стороннього спостерігача процес сприймався мов звичайний обмін цифровими файлами. Задля того, аби зробити систему більш безпечною, дані треба шифрувати перед впровадженням стегографії.

Цим способом, в даному в проекті буде реалізована стегосистема, що дозволить вирішити наступні проблеми:

- надійно приховувати і відновлювати дані, керувати цим процесом поза поміччю зручного візуального інтерфейсу;
- вивчати і аналізувати основні моделі і принципи цифрової стегографії;
- вивчати властивості і особливості форматів, котрі будуть контейнерами задля конфіденційної даних.

1.6 Аналіз структури стеганографічної системи

Стеганографічна система (стегосистема) реалізує задачу вбудовування та виділення сповіщення із бокса. Стегосистема складається із основних складових, показаних на рис. 1.10.

Вбудовування повідомлень у бокс проходить із використанням спеціального стегоключа. Код – псевдовипадкова послідовність біт, яку створює генератор, що задовольняє певним вимогам (криптографічно безпечний генератор). Цей код визначає порядок вбудовування сповіщення у бокс. У якості основи генератора спроможне використовуватися, наприклад, лінійний рекурентний регістр. Тоді адресатам задля забезпечення зв'язку спроможне повідомлятися початкове результати цього регістра.

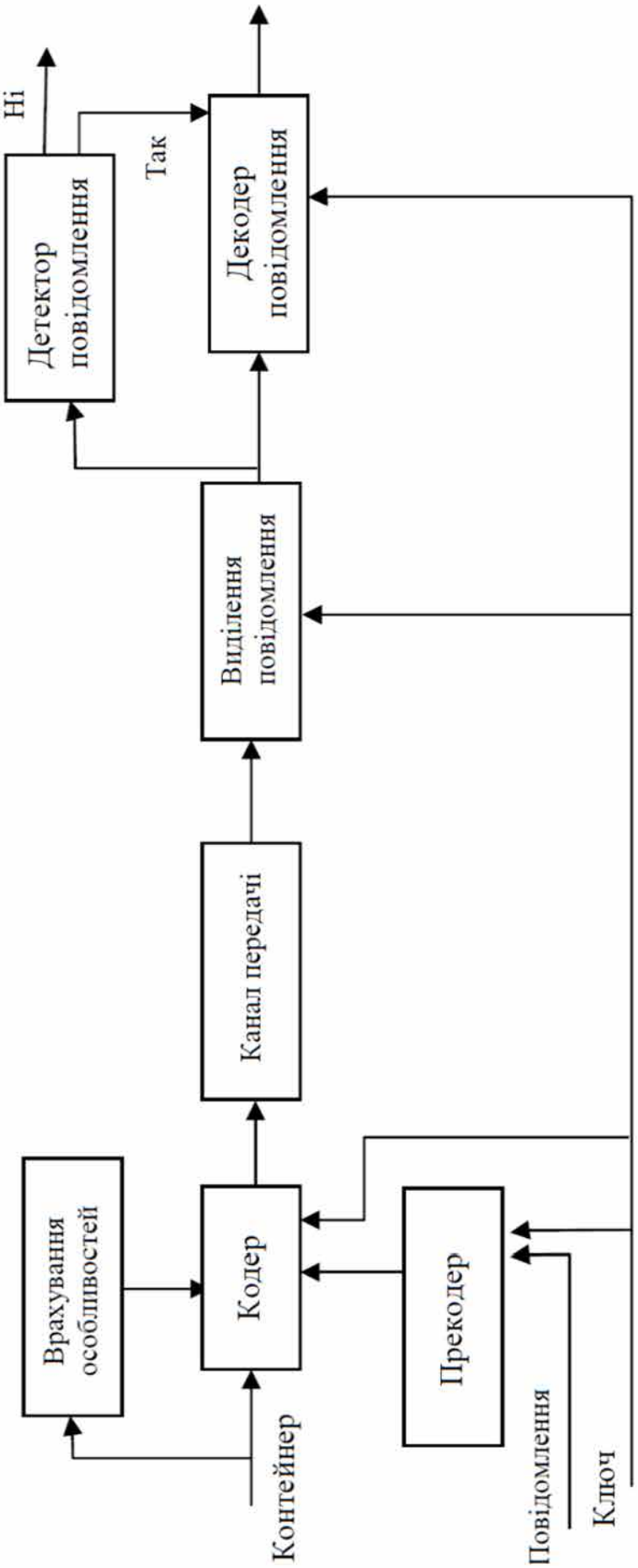


Рисунок 1.10. Структурна схема типової стего-системи

Таємна інформація вбудовується в відповідності щодо ключа у ті відліки, спотворення котрих не призводить щодо суттєвих спотворень бокса. Ці біти утворюють стегошлях. Під суттєвим спотворення треба розуміти спотворення, яке призводить мов щодо неприйнятності задля людини-адресата заповненого бокса, так та щодо можливості виявлення факту наявності таємного сповіщення опісля стегоаналізу.

1.7 Впровадження технологій стегографії і зашифровування в проекті

В цьому підрозділі будуть проаналізовані основні технології та базові алгоритми задля реалізації поставленої задачі по створенню програмного застосунку задля передавання та захищення схованих інформації. Перш поза все будуть застосовані метод ISB, алгоритм зашифровування A-E-S, стиснення JPG.

1.7.1 Впровадження способу найменш значущого біта (ISB)

Метод заміни найменш значущого біта (Least Significant Bits – ISB) задля графічних контейнерів полягає у приховуванні даних шляхом зміни останніх біт фото, котрі кодують колір, на біти приховуваного сповіщення. Різниця поміж порожнім та заповненим контейнерами повинна бути не відчутна задля органів сприйняття людини. Принцип приховання даних показано на рис. 1.11, рис. 1.12.

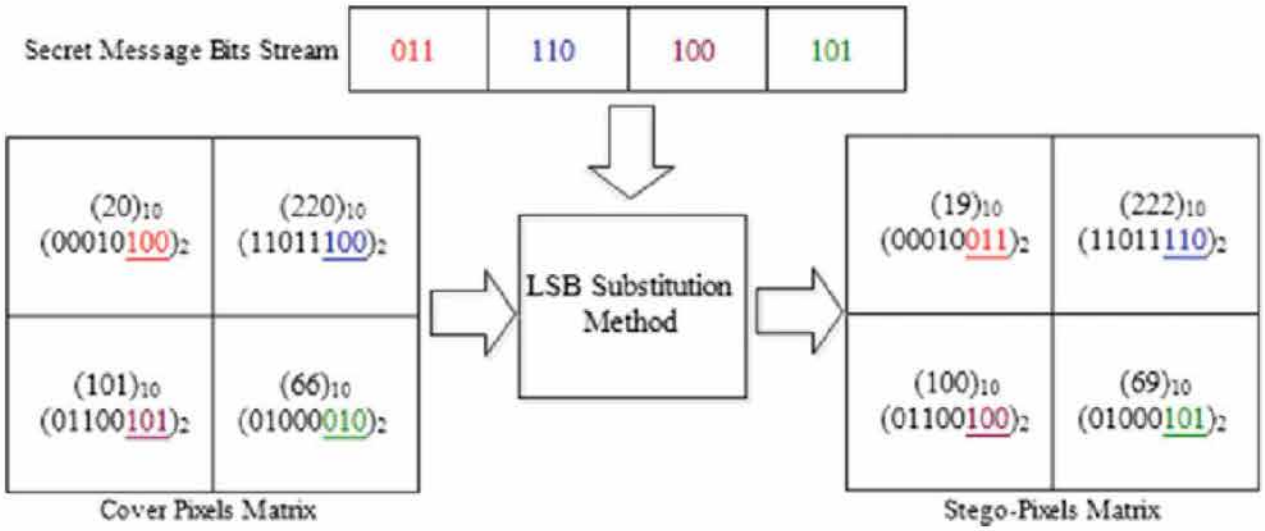


Рисунок 1.11. Принцип приховання даних в зображенні поза методом ISB

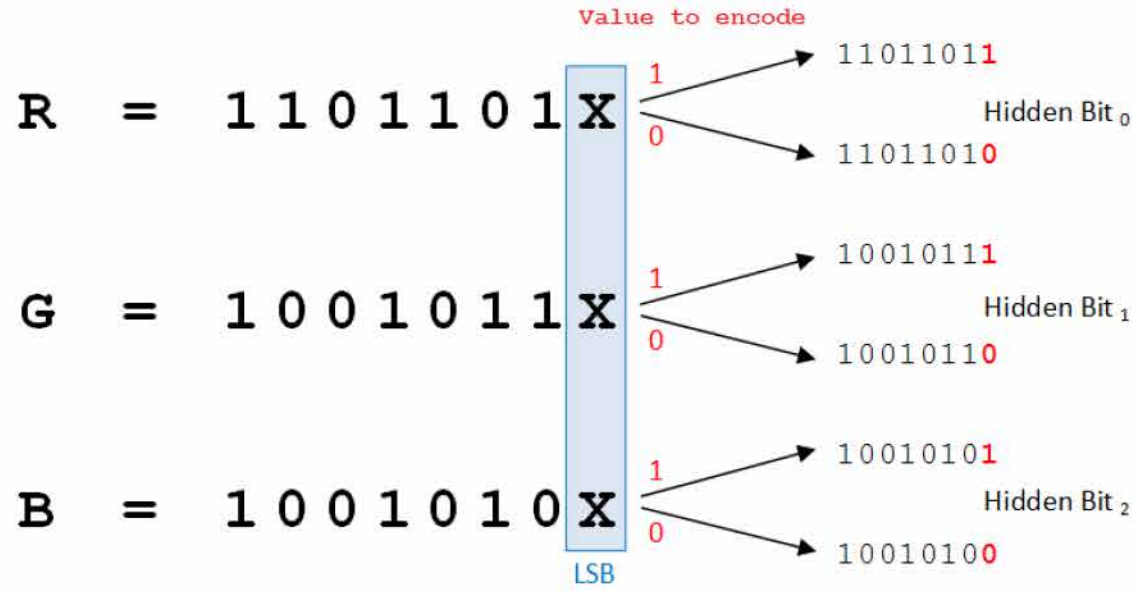


Рисунок 1.12. Принцип кодування схованих біт поза методом ISB

В форматі BMP фото зберігається мов матриця значень відтінків кольору задля кожної точки фото, що зберігається. Коли кожна із компонент простору RGB (їх ще називають каналами кольору) зберігається у одному байті, вона спроможне набувати значень з О щодо 255 включно, що надає 24-бітній глибині кольору. Особливість зору людини полягає у тому, що слабо розрізняються незначні коливання кольору. Задля 24-бітного кольору зміна у кожному із трьох каналів одного найменш значимого біта (тобто крайнього правого) призводить щодо зміни менш ніж на 1% інтенсивності даної точки, що надає змінювати їх непомітно задля ока.

Коли не враховувати зазвичай незначну поза розміром службову дані на початку файлу фото (рис. 1.13), буде можливість потай передати сповіщення розміром у 1/8 розміру бокса ("розмазати" поза останніми бітам у кожному байті матриці кольорів пікселів) чи ж розміром у 1/4 розміру бокса (відповідно у використанні двох останніх біт).

Стеганографічний метод полягає у наступному. Нехай існує 24-бітове фото у градаціях сірого. Піксель кодується 3 байтами та у них розташовані результати каналів RGB. Змінюючи найменш значущий біт ми міняємо результати байта на одиницю. Ці градації, мало того що непомітні задля людини, можуть взагалі не відобразитися у використанні низькоякісних пристроїв виведення.

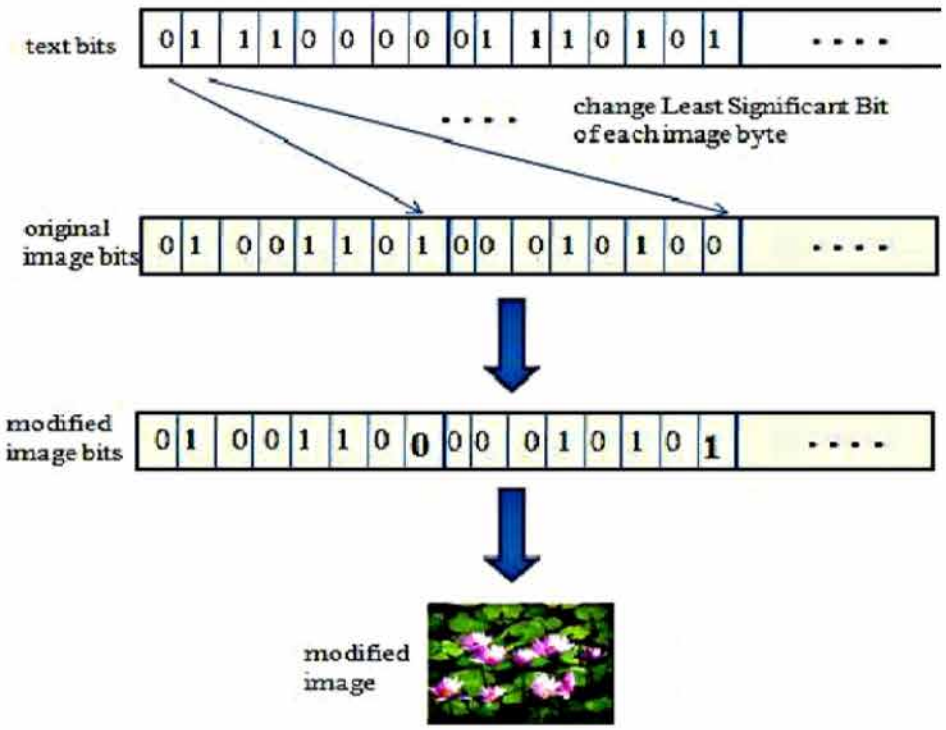


Рисунок 1.13. Приховання даних у зображенні поза методом ISB

Приклад, наведений нижче, показує, мов сповіщення спроможне бути приховано у перших восьми байтах, що відносяться щодо трьох пікселів в зображенні із глибиною кольору 24-біт:

Кодування пікселів фото:

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

Двійковий код схованої даних: 01000001

Формування результату:

(0010011 0 1110100 1 1100100 0)

(0010011 0 1100100 0 1110100 0)

(1100100 0 0010011 1 1110100 1)

Тут виділені і відокремлені три біти, котрі були фактично змінені.

Впровадження стеганографічного способу ISB у середньому вимагає, аби тільки половина біт фото-бокса були змінені. Невелика модифікація цієї стеганографічної техніки надає застосовувати задля вбудовування сповіщення два чи більш молодших біт на байт. Це збільшує обсяг схованої даних у об'єкті-

контейнері, але скритність сильно знижується, що полегшує процес розпізнавання стегографії. Інші варіації цього способу включають у себе нівелювання статистичних змін у зображенні. Існує інтелектуальне програмне забезпечення задля стеганоаналізу, яке перевіряє області, що складаються із одного суцільного кольору. Задля підвищення скритності слід уникнути запису змін у ці пікселі. Моделі ISB є нестійкими щодо всіх видів атак та можуть бути використані тільки у відсутності шуму у каналі передавання інформації. Виявлення ISB-кодованого бокса здійснюється поза аномальними характеристиками розподілу значень діапазону молодших біт відліків цифрового сигналу.

Цим способом, сутність способу найменш значущого біта (ISB) полягає у тому, що поза поміччю молодшого біта чи біт здійснюється стегано-графія сповіщення. Задля реалізації стегографії сповіщення буде перетворено в двійковий формат, тоді заголовок файлу буде опущено, але у метаданих застосовується алгоритм ISB. Опісля перетворення фото на двійкове представлення останній біт кожного байту замінюється бітом в двійковій послідовності. У результаті буде отримано фото, яке треба передати по відкритому каналу та зловмисник не зможе здогадатися про наявність конфіденційної даних. Опісля отримання бокса одержувач спроможне легко витягнути сповіщення із фото.

1.7.2 Впровадження алгоритму зашифровування A-E-S

Алгоритм зашифровування A-E-S (стандартизований алгоритм Rijndael) є симетричним алгоритмом зашифровування (розмірність блоку 128 біт, код 128/192/256 біт). Поза алгоритмом A-E-S 128 біт даних розбиваються на байти і упаковуються у масив 4x4.

Шифр A-E-S є алгоритмом блочного зашифровування із відкритим ключем, що використовує 128-бітні дані і ключі різної довжини.

Масив байтів записуються в таблицю, де виконуються стандартні математичні операції над окремими елементами набору (рис. 1.14).

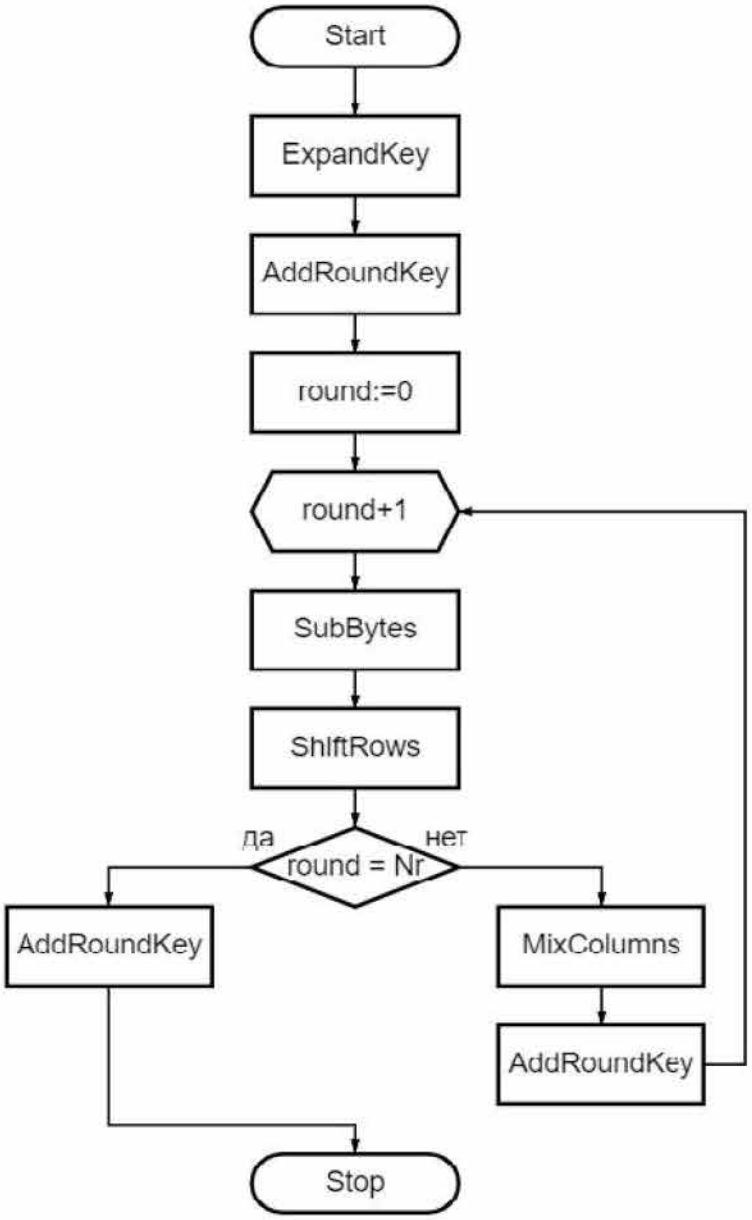


Рисунок 1.14. Реалізація зашифровування поза алгоритмом A-E-S

Задля кожної ітерації даного способу обирається свій підключ. Функції, що застосовуються в даному алгоритмі:

- SubBytes – таблична заміна кожного байту. На відміну з DES таблиця відповідностей не змінюється;
- ShiftRows – циклічний зсув вліво;
- MixColumns – операції поліноміальної арифметики чи дії над полем Галуа $GF(2^8)$;
- AddRoundKey – побітова операція XOR із відповідним байтом ключа ітерації.

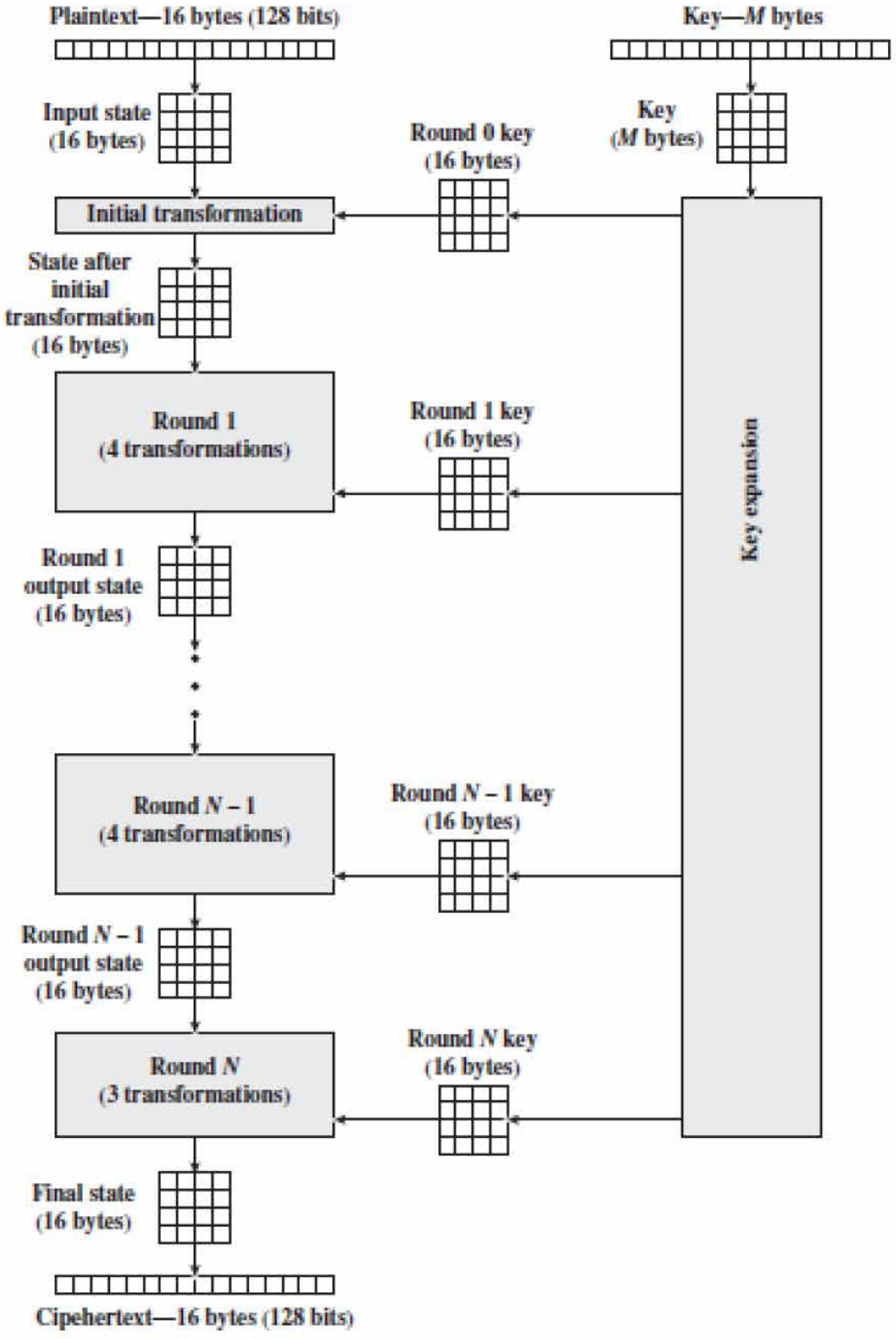


Рисунок 1.15. Схема роботи алгоритму A-E-S

Перевагою впровадження даного способу є швидкість, оскільки він оперує байтами і реалізація відбувається на процесорі. Зазвичай на цей алгоритм здійснюються атаки сторонніми каналами. Недоліками є:

– однакове зашифровування задля ідентичних інформації (кожний блок завжди кодується однаково);
– проста структура;
– важка імплементація поза поміччю програмного забезпечення.

Шифр Rijndael, на базі якого побудовано стандартизовану версію A-E-S, не є точно цим самим. Шифр Rijndael підтримує три різні розміри блоків: 128, 192 і 256 біт. Однак стандартний шифр A-E-S визначає розмірність блоку лише 128 біт. Шифр A-E-S підтримує три різні довжини ключів: 128, 192 та 256 біт.

В шифрі A-E-S із розміром блоку 128 біт структура функції зашифровування розглядається мов мережа заміщення-перестановки. Структура такої мережі проілюстрована на рис. 1.15.

128-бітні раундові ключі виробляються розкладанням ключів із головного ключа. Раундові ключі змішуються в раундовій функції із станом блоку поза поміччю XOR. Раундова функція реалізує фазу заміщення, передаючи шістнадцять байт стану блоку через 8-бітну функцію S-боксів. S-поле A-E-S реалізує мультиплікативне обернення, зокрема, перетворення кінцевого поля GF (28) в вигляді множення бітової матриці. Мов результат, функція S-боксів є бієкцією, необхідною задля побудови мережі перестановки. Зворотний блок S задля дешифрування побудований із оберненої трансформації і тієї ж мультиплікативної інверсії у GF (28). Фаза заміщення дешифрування, що реалізує шістнадцять паралельних зворотних операцій S-боксів, називається InvSubBytes.

Фаза перестановки A-E-S розділена на дві різні фази – ShiftRows і MixColumn. Задля цих фаз стан блоку треба розглядати мов 4 × 4-байтну матрицю. ShiftRows обертає результати у матриці стану на різні величини вздовж рядків. Перший ряд не обертається (чи треба сказати, що він обертається на нульові місця), другий ряд – на одне місце, третій – на два, але останній ряд – на три місця. Фаза MixColumn реалізує множення матриць в конкретному кінцевому полі GF (28) із кожним стовпцем окремо мов вхід та вихід. Обидві ці операції можуть бути змінені, та ці звороти, котрі використовуються у дешифруванні, називаються InvShiftRows і InvMixColumns.

Процес зашифровування A-E-S починається із початкової трансформації, що є додатковою операцією AddRoundKey. Дешифрування треба виконати, змінивши порядок.

W1	W2	W3	W4	W5	W6	W7	W8								
2B	28	AB	09	A0	88	23	2A								
7E	AE	F7	CF	FA	54	A3	6C	.....				.....			
15	D2	15	4F	FE	2C	39	76								
16	A6	88	3C	17	B1	39	05								
Cipher key				Round key 1				.....				Round key 10			

RW		W1		S1		R1		W5
CF		2B		8A		01		A0
4F		7E		84		00		FA
3C	⊕	15	⊕	EB	⊕	00	=	FE
09		16		01		00		17

Рисунок 1.16. Фаза отримання раундових ключів

Раундові ключі отримуються у блоці розширення ключів. Код шифру вміщує довжину 128 біт та упакований в масив 4x4, мов показано на рис.1.16. Аби отримати W5, спочатку береться W1, потім реалізується операція RotWord, що переміщує стовпець вгору, але 1-й елемент опускається, закінчуючи RW. Потім реалізується операція SybeBytes із цим стовпцем, у результаті чого виходить S1. W5 отримано є логічним добутком W1, S1 та R1 (масив Rcon, котрий треба знайти у в документації шифру A-E-S) в стовпці 1. Решта складових отримується цим способом: W6 = W2 ⊕ W5, W7 = W3 ⊕ W6, W8 = W4 ⊕ W7. Решту фазових ключів отримують аналогічно, але початковий код вибирають із останнього отриманого. Зашифровування і дешифрування відбувається поза поміччю процедур AddRoundKey, SubBytes, ShiftRows, MixColumns, AddRoundKey, InvSubBytes, InvShiftRows, InvMixColumns.

В процедурі SubBytes перша половина байтового набору наводить на рядок, але друга половина наводить на стовпець. Потім реалізується заміна результати набору на результати рядка і стовпця, отриманих вище в SBox. Підстановка у

InvSubBytes реалізується цим самим способом, але результати рядка береться із набору InvSbox (рис. 1.17).

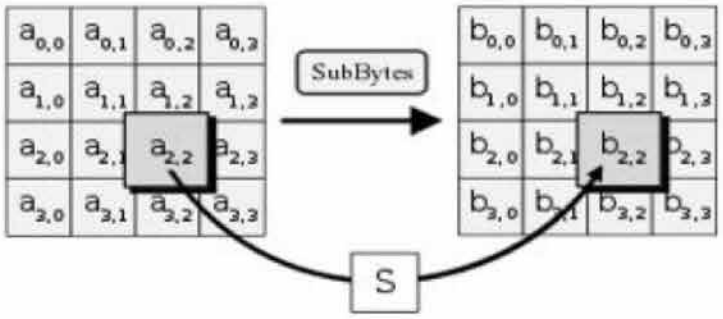


Рисунок 1.17. Реалізація фази SubBytes поза алгоритмом A-E-S

В процедурі SubBytes перша половина байтового набору наводить на рядок, але друга половина наводить на стовпець. Потім реалізується заміна результати набору на результати рядка і стовпця, отримані вище у SBox. Підстановка у InvSubBytes реалізується цим же способом, результати рядка береться із набору InvSbox (рис. 1.18).

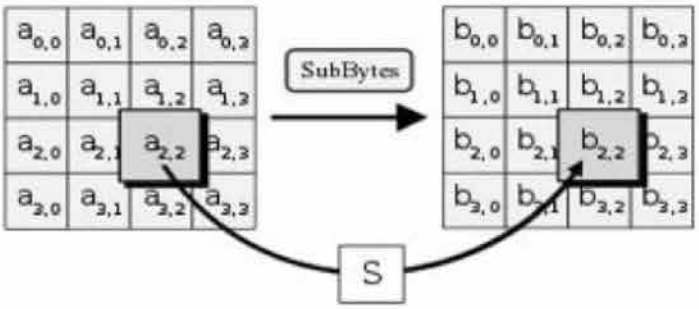


Рисунок 1.18. Реалізація фази SubBytes поза алгоритмом A-E-S

На етапі ShiftRows результати у першому рядку не зсуваються, в 2-му рядку зсуваються на 1 вліво, у 3-му – на 2, у 4-му – на 3, де результати із першого стовпця переміщується у кінець. InvShiftRows разом з цим реалізується в зворотному порядку (рис. 1.19).

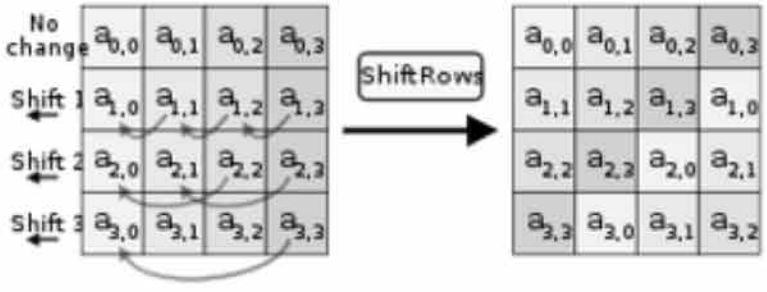


Рисунок 1.19. Реалізація фази ShiftRows поза алгоритмом A-E-S

Процедура MixColumns бере чотири байти кожного стовпця набору і множить його на фіксований поліном $c(x) = 3x^3 + x^2 + x + 2$ поза модулем GF(28) $x^4 + 1$. В InvMixColumns множення реалізується на фіксований поліном $c(x) = 11x^3 + 13x^2 + 9x + 14$ (рис. 1.20).

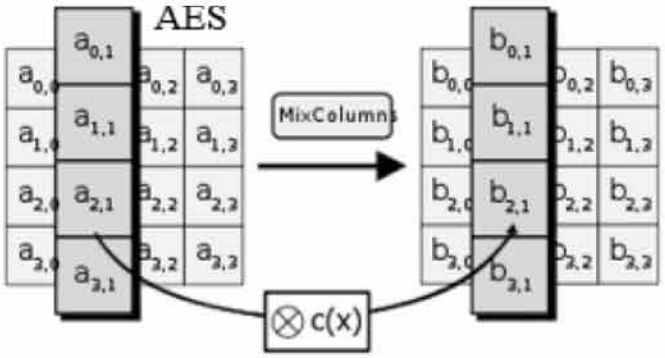


Рисунок 1.20. Реалізація фази MixColumns поза алгоритмом A-E-S

На етапі AddRoundKey результати у масиві логічно перемножуються (XOR) на відповідний елемент фазового ключа відповідно (рис. 1.21).

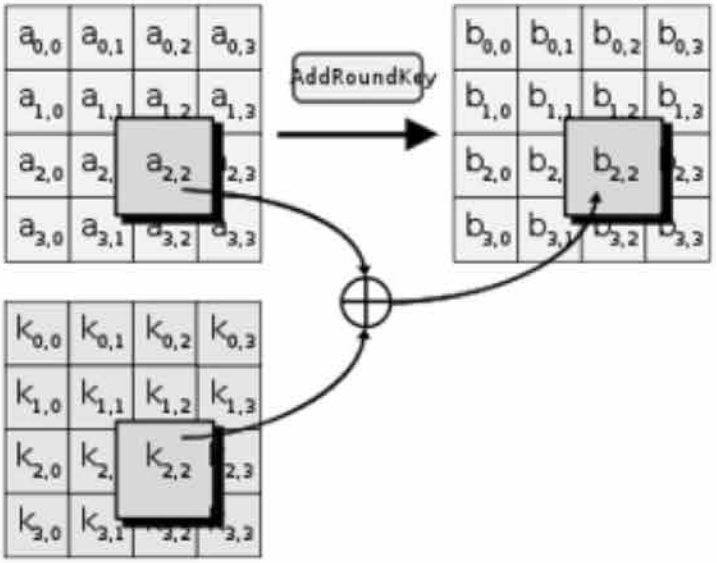


Рисунок 1.21. Реалізація фази AddRoundKey поза алгоритмом A-E-S

У даний час криптографія A-E-S широко прийнята та підтримується мов у апаратному, так та у програмному забезпеченні. Щодо теперішнього часу жодної практичної криптоаналітичної атаки проти A-E-S не виявлено. Крім того, A-E-S вміщує вбудовану гнучкість довжини ключів, що надає певним способом захищати майбутнє з прогресу в можливості виконувати вичерпні пошуки ключів.

1.7.3 Впровадження формату PNG

Формат PNG (Portable Network Graphics) є одним із найпопулярніших форматів веб-графіки. Поза цим форматом фото піддається стисненню без втрат. Формат PNG підходить задля зберігання проміжних версій фото. Він підтримує велику число кольорів, зокрема PNG8 – 256 кольорів та PNG24 – близько 16,7 мільйонів кольорів. В форматі передбачено можливість додавання мета-інформації щодо файлу (у необхідності захист авторських прав). Це формат, котрий надає отримувати фото із прозорим фоном (наприклад, це дуже важливо у створенні логотипів, задля котрих зазвичай потрібен прозорий фон). Щодо недоліків – немає підтримки анімації, неможливо зберегти кілька фото у одному файлі. Крім логотипів, цей формат разом з цим застосовується задля створення складових навігації сторінок, типографіки, літографій, текстів, малюнків із чіткими краями фото.

Оскільки формат PNG зберігає фото без втрати кольору, стегано-графія реалізується у молодших бітах пікселів фото поза поміччю розглянутого вище алгоритму ISB (найменш значущого біту).

1.7.4 Впровадження формату BMP

Формат BMP (BitMap, растрове фото чи бітовий масив) є нестиснутим форматом растрової графіки задля зберігання фото, розробленим Microsoft. ІЗ форматом BMP працює велика число програм, адже його підтримка інтегрована у операційні системи Windows. ОС Windows вміщує спеціальні функції API, котрі допомагають читати і відображати фото формату BMP. Структура BMP складається із чотирьох частин: BitMapFileHeader; BitMapInfoHeader; таблиця кольорів; масив BitMap.

Оскільки формат BMP зберігає фото без втрати кольору, стегано-графія реалізується у молодших бітах пікселів фото поза поміччю розглянутого вище алгоритму ISB (найменш значущого біту).

1.7.5 Впровадження формату JPG

Формат JPG є одним із популярних графічних форматів растрової графіки, котрий застосовується задля зберігання фото. Файли, що містять дані JPG, зазвичай мають розширення .jpg, .jfif, .jpe чи .jpg. Однак .jpg є найпопулярнішим із них на всіх платформах. Файл JPG вміщує послідовність вказівників, кожен із котрих починається із байту Oxff, що наводить на початок вказівника. Деякі маркери складаються лише із цієї пари байтів, тоді мов інші містять додаткові дані, що складаються із двох байтів, котрі вказують на довжину вказівника, включаючи байти довжини. Така файлова структура надає швидко знайти вказівник із необхідними даними (наприклад, довжина рядка, число рядків та число колірних компонентів стиснутого фото). Основні вказівники JPG:

– Oxff 0xD8 SOI – початок закодованої частини фото;

– Oxff 0xC0 SOF0 – наводить на те, що фото закодовано у базовому режимі поза поміччю кодів DCT та Хафмана. Вказівник вміщує довжину і ширину фото, число компонентів (рівно 8 біт задля кожного компонента) та співвідношення сторін компонентів (наприклад, 4:2:O);

– Oxff 0xC1 SOF1 – наводить, що фото закодовано у розширеному режимі із використанням кодів DCT та Хафмана. Вказівник вміщує довжину та ширину фото, число компонентів (число біт на компонент 8 чи 12) та співвідношення компонентів (наприклад, 4:2:O);

– Oxff 0xC2 SOF2 – наводить, що фото закодовано у прогресивному режимі із використанням кодів DCT та Хафмана. Вказівник вміщує довжину та ширину фото, число компонентів (число біт на компонент 8 чи 12) та співвідношення компонентів (наприклад, 4:2:O);

– Oxff 0xC4 DHT – цей вказівник вміщує одну чи більше таблиць Хафмана;

– Oxff 0xDB DQT – всередині цього маркера є одна чи більше таблиць дискретизації;

– Oxff 0xDA SOS – початок першого чи наступного сканування фото зліва направо, зверху вниз. Коли ми маємо справу із базовим чи розширеним режимом

кодування, застосовується одне сканування. В прогресивному режимі застосовується кілька сканерів. Можливо, маркер SOS розділяє інформаційну (заголовок) та зашифровану (стиснуті дані фото) частини фото;

– Oxff 0xFE COM – маркер коментаря;

– Oxff 0xD9 EOI – кінець закодованої частини фото.

Формат JPG, на відміну з форматів BMP та PNG, зберігає фото із втратою кольору. ІЗ цієї причини ми не можемо виконати стеганографію у пікселях поза поміччю способу ISB, котрий ми можемо застосовувати в форматах PNG та BMP. Аби виконати стегано-графія, треба розуміти структуру формату JPG.

FF	D8	FF	FE	00	04	3A	29	FF	DB	00	43	00	A0	6E	78
8C	78	64	A0	8C	82	8C	B4	AA	A0	BE	F0	FF	FF	F0	DC
DC	F0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	DB	00
43	01	AA	B4	B4	F0	D2	F0	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	C0	00	11	08	00	10	00	10	03	01	22	00	02
11	01	03	11	01	FF	C4	00	15	00	01	01	00	00	00	00
00	00	00	00	00	00	00	00	00	00	03	02	FF	C4	00	1A
10	01	00	02	03	01	00	00	00	00	00	00	00	00	00	00
00	01	00	12	02	11	31	21	FF	C4	00	15	01	01	01	00
00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	FF
C4	00	16	11	01	01	01	00	00	00	00	00	00	00	00	00
00	00	00	00	11	00	01	FF	DA	00	0C	03	01	00	02	11
03	11	00	3F	00	AE	E7	61	F2	1B	D5	22	85	5D	04	3C
82	C8	48	B1	DC	BF	FF	D9								

Рисунок 1.22. Іконка сайту Google в шістнадцятковій СЧ

Існує 3 типи формату JPG: базовий, розширений та прогресивний. В даному проекті буде використовуватись стегано-графія в базовому режимі.

Фото розділене на вказівники, довжина котрих становить 2 байти, але перший байт – [FF]. Майже всі вказівники зберігають свою довжину у наступних 2 байтах опісля вказівника. На рис. 1.22 показане кодування іконки сайту Google в 16-й системі. Маркери пофарбовані в синій колір.

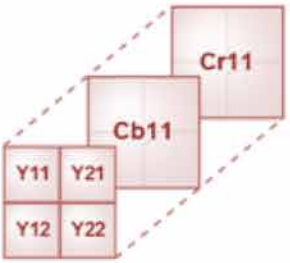


Рисунок 1.23. Стиснення каналів Cb та Cr в 2 рази у JPG-форматі

Алгоритм стиснення в форматі JPG передбачає 6 основних етапів:

– фото перетворюється із колірної гами RGB на гамму YCbCr;

– часто канали Cb та Cr стискаються у 2 рази. В цьому разі кожен потік Y отримує усереднене результати каналів Cb та Cr (рис. 1.23);

– результати каналів розбиваються на блоки 8x8;

– кожен блок піддається дискретному косинусному перетворенню. Опісля цих дій буде отримано масив масштабів 8x8. Множник у лівому куті набору називається DC (він є найважливішим коефіцієнтом та вважається середнім значенням всіх значень), але інші 63 результати називаються AC;

– отримані коефіцієнти квантуються, кожен із них множиться на складові набору дискретизації (кожен потік вміщує власний масив дискретизації);

– на завершальному етапі отримані масиви кодуються методом Хафмана. Закодовані масиви зберігаються у порядку Yi Cbi Cri, наприклад, коли канали Cb та Cr стискаються 2 рази Y00 Y10 Y01 Y11 Cb00 Cr00 Y20... (рис. 1.24).

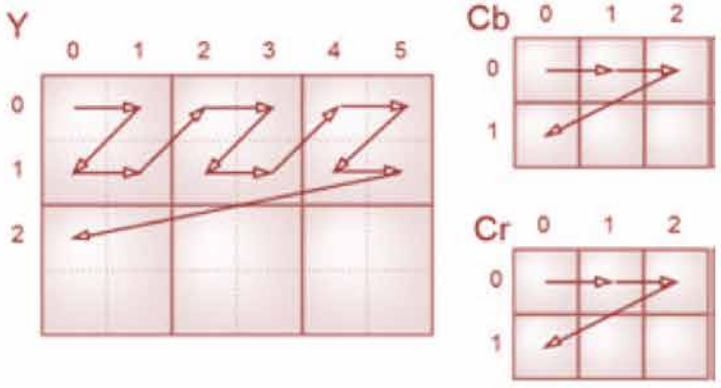


Рисунок 1.24. Розташування каналів Y, Cb, Cr у JPG-форматі

Маркери, поза поміччю котрих буде здійснюватися стегано-графія, мають наступний вигляд:

– Oxff 0xC0 SOF0 – базовий DCT;

– Oxff 0xDB DQT – таблиця дискретизації;

– Oxff 0xC4 DHT – таблиця Хафмана;

– Oxff 0xDA SOS (Start of Scan) – початок сканування.

Базовий DCT-маркер SOF0 означає, що фото закодовано базовим методом:

– [00 11] Довжина. 17 байт;

– [08] Точність. 8 біт. В цьому режимі воно завжди дорівнює 8. Наводить бітрейт каналу;

– [00 10] Висота фото. 0x10 = 16;

– [00 10] Довжина фото. 0x10 = 16;

– [03] Число каналів. 3. Переважно Y, Cb, Cr чи R, G, B.

1-й потік:

– [01] індентифікатор. 1;

– [2_] Горизонтальне проріджування (H1). 2;

– [_2] Вертикальне проріджування (V1). 2;

– [00] Визначник таблиці дискретизації. O.

2-й потік:

– [02] визначник. 2:

– [1_] Горизонтальне проріджування (H2). 1;

– [_1] Вертикальне проріджування (V2). 1;

– [01] Визначник таблиці дискретизації. 1.

3-й потік:

– [03] визначник. 3;

– [1_] Горизонтальне проріджування (H3). 1;

– [_1] Вертикальне проріджування (V3). 1.

Визначаємо Hmax = 2 та Vmax = 2. Потік та буде стиснутий в Hmax / Hi разів по горизонталі і в Vmax / Vi разів по вертикалі.

В DQT-таблиці дискретизації:

– [00 43] Довжина вказівника. 0x43 = 67 байтів;

– [O_] Довжина результати. O (O – 1 байт, 1 – 2 байти);

– [_O] Визначник таблиці. O.

Інші 64 байти потрібно заповнити у масив 8x8. Результати заповнюються у масиві поза поміччю способу зигзагоподібного порядку, мов показано на рис. 1.25.

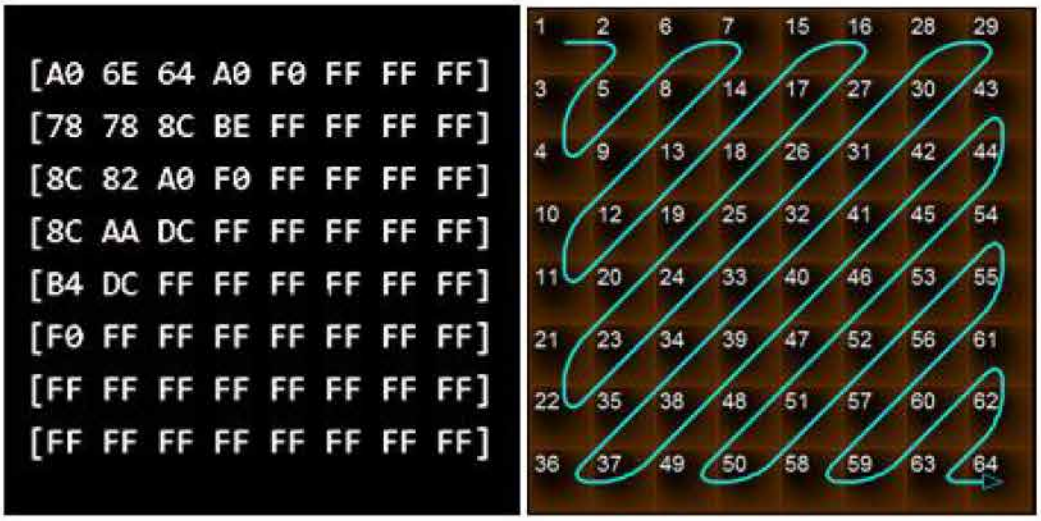


Рисунок 1.25. Впровадження способу зигзагоподібного порядку байтів

В DHT-таблиці Хафмана зберігаються результати і коди, закодовані кодом Хафмана:

– [00 15] Довжина покажчика. 21 байт;

– [O_] Наводить на множник таблиці. (O – множник DC, 1 – множник AC);

– [_O] Визначник таблиці. O.

Наступні 16 значень є такими:

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

Число кодів [01 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00].

У цьому розділі зберігаються лише довжини кодів, але коди потрібно шукати окремо. Отже, є один код довжини 1 та один код довжини 2.

Результати мають довжину у один байт, тому зчитується 2 байти:

– [03] – перше результати коду;

– [02] – друге результати коду.

Передбачено ще 3 маркери Хафмана, котрі розшифровуються аналогічно.

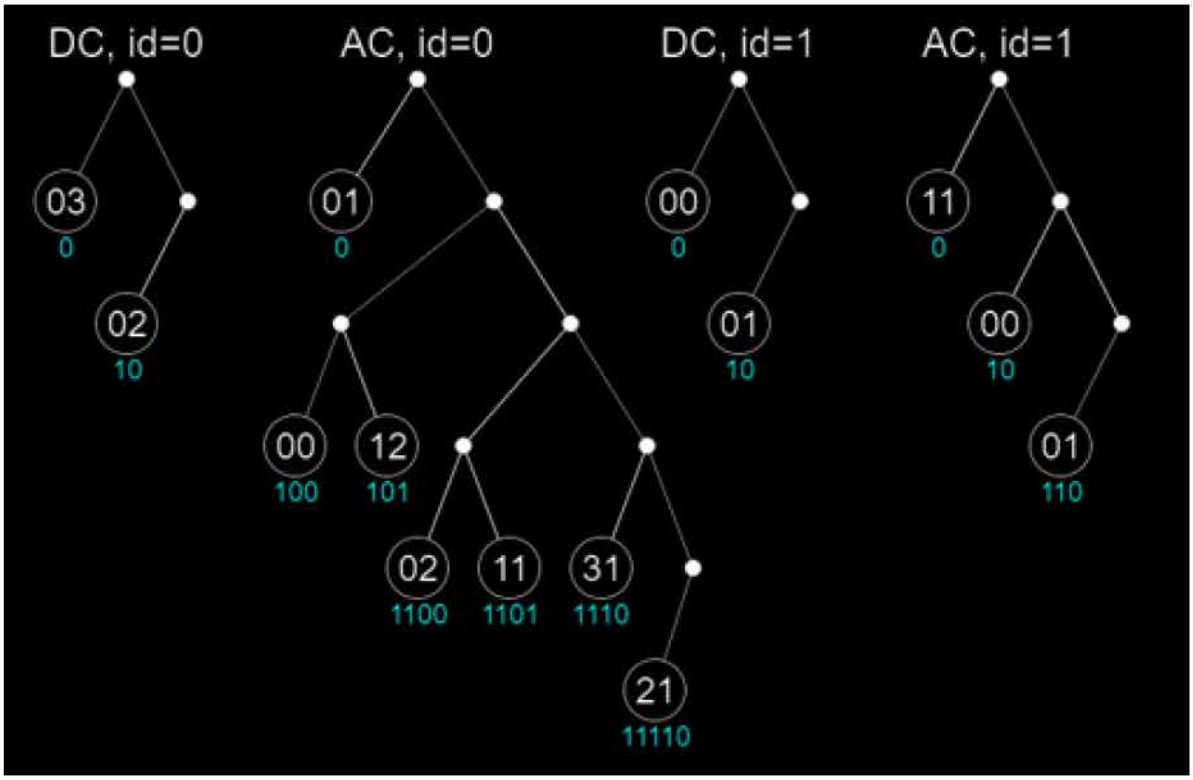


Рисунок 1.26. Впровадження дерева Хафмана

Маючи всі таблиці Хафмана, треба поза поміччю отриманих таблиць побудувати бінарне дерево. Побудувавши дерево, треба декодувати коди. Отримавши коди, їх треба розмістити в таблиці 8x8 зигзагоподібно. Коли на місці немає результати, треба повертатись на вищий рівень. Зупинитись потрібно на рівні, рівному довжині коду. Ліві гілки відповідають О, але праві – 1. На рис. 1.26 вказані результати, отримані у таблицях, але разом з цим коди, якими кодується фактичний вміст кодованого фото.

В SOS-таблиці (Start of Scan) позначається початок сканування:

– [00 0C] Довжина. 12 байт;

– [03] число каналів. 3 канали: Y, Cb, Cr.

1-й потік:

– [01] визначник каналу. 1 (Y);

– [O_] визначник таблиці Хафмана задля DC-масштабів: O;

– [_O] визначник таблиці Хафмана задля AC-масштабів: O.

2-й потік:

– [02] визначник каналу. 2 (Cb);

– [1_] визначник таблиці Хафмана задля DC-масштабів: 1;

– [_1] визначник таблиці Хафмана задля AC-масштабів: 1.

3-й потік:

– [03] визначник каналу. 3 (Cr);

– [1_] визначник таблиці Хафмана задля DC-масштабів: 1;

– [_1] визначник таблиці Хафмана задля AC-масштабів: 1.

[00], [3F], [00] – Start of spectral or predictor selection, End of spectral selection, Successive approximation bit position. Ці результати використовуються тільки задля прогресивного режиму. Звідси щодо кінця (вказівник [FF D9]) кодуються дані. Перших 33 біт буде достатньо задля побудови першого набору масштабів.

Задля знаходження DC-масштабів:

1. Зчитується послідовність біт, але коли зустрічається 2 байти [FF 00], то це не вказівник, але просто байт [FF]. Опісля кожного біта треба рухатися по дереву Хафмана (із відповідним ідентифікатором). В разі 0 треба рухатися вздовж гілки вліво, але в разі 1 – вправо та так далі, поки не буде досягнуто кінцевого вузла

10101110111001110110000111110010 0

2. Треба взяти результати вузла. Коли воно дорівнює 0, то множник дорівнює 0, воно записується щодо набору та продовжується пошук інших масштабів. В показаному разі 02 наводить на довжину коефіцієнта. Тобто відбувається зчитування наступних 2 біт, котрі будуть коефіцієнтом:

10101110111001110110000111110010 0

1. Коли перша цифра результати дорівнює 1 в двійковій системі, то не треба робити зміну DC = <результати>. У іншому разі реалізується така операція DC = <результати> – 2 ^ <довжина результати> + 1.

DC-множник зчитується лише один раз, опісля чого зчитуються AC-коефіцієнти:

1. Подібно щодо знаходження коефіцієнта DC продовжує зчитуватися послідовність

10101110111001110110000111110010 0

2. Треба взяти результати вузла. Коли він дорівнює О, це означає, що решту значень матриці потрібно заповнити нулями. Наступні байти кодують наступну матрицю. В даному разі результати вузла дорівнює 0x31:

а) Результати першої половини байту наводить, скільки нулів потрібно додати щодо матриці. В даному разі треба додати три нулі:

b) Результати другої половини байту наводить на довжину коефіцієнта. В даному разі 1, тобто наступний біт – це множник.

10101110111001110110000111110100

3. Зчитувати множник АС треба щодо тих пір, поки не зустрінеться нульове результати вузла чи поки матриця не заповниться. Цим способом, масив масштабів матиме такий вигляд:

10101110111001110110000111110100

$$\begin{bmatrix} 2 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Аналогічно будуть отримані інші три результати Y. Але задля них коефіцієнти DC є різницею поміж коефіцієнтами DC попередніх масивів:

– DC задля 2-го набору. 2 + (-4) = -2;

– DC задля 3-го набору. -2 + 5 = 3;

– DC задля набору 4. 3 + (-4) = -1.

Канали Cb та Cr схожі. Розглянуто тільки канали Y1 та Cb, Cr:

$$\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

На етапі дискретизації складові набору треба помножити на складові

набору дискретизації. Мов із'ясовано із блоку S0F0, Y-результати множаться на масив із ідентифікатором О, але Cb, Cr на масив із ідентифікатором 1 відповідно. Тож опісля реалізація множення буде отримано 4 масиви Y та масиви Cb, Cr. Розглянемо лише результати Y1 та Cb, Cr. Опісля етапу дискретизації всі коефіцієнти DC підсумовуються щодо 1024 значень:

$$\begin{bmatrix} 1344 & 0 & 300 & 0 & 0 & 0 & 0 & 0 \\ 0 & 120 & 280 & 0 & 0 & 0 & 0 & 0 \\ 0 & -130 & -160 & 0 & 0 & 0 & 0 & 0 \\ 140 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 854 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 180 & 210 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1024 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 180 & -210 & 0 & 0 & 0 & 0 & 0 & 0 \\ 240 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Опісля виконаних дій отримано обернене дискретне косинусне перетворення (1.1).

$$S_{yx} = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C_u C_v S_{vu} \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16} \qquad (1.1)$$

де Y та X – номери стовпців та рядків коефіцієнта, U та V є змінними, котрі змінюються у межах О-7, коли U дорівнює О, тоді Cu=1/√2, інакше Cu = 1, коли V дорівнює О, тоді Cv=1/√2, інакше Cv = 1, Svu – необхідний множник.

Стиснення проводилося із співвідношенням Cb:Y, Cr:Y 2:1 (оскільки не заповнювався SOF0 – базовий блок DCT), тому кожен піксель описується формулою pixel (Y_{ij}, $Cb_{[i/2,j/2]}$, $Cr_{[i/2,j/2]}$).

Останнім етапом є перехід з YCbCr щодо системи RGB:

1. R=Y+1,402*Cr;

2. G=Y-O,34414*Cb-O,71414*Cr;

3. B=Y+1,772*Cb.

У виході значень поза діапазон [O, 255] треба брати граничні результати. У результаті буде отримано таблиці каналів R, G, B верхнього лівого квадрата 8x8 даного фото.

У результаті всіх вище виконаних дій фото JPG перетворюється поза базовим форматом. В розглянутому прикладі були знайдені коефіцієнти поза поміччю дерева Хафмана. Поза поміччю цих масштабів здійснюється стеганографія. Молодші розряди масштабів замінюються бітами секретної даних. Спотворення залежить з того, скільки біт застосовується із кінця коефіцієнта. Множник DC не змінюється, оскільки він спричиняє значні спотворення фото.

1.7.6 Впровадження аудіоформату WAVE

Нестиснені звукові файли із розширенням WAV чи WAVE (Waveform Audio file) є стандартним форматом задля зберігання оцифрованої музики і звуку. Найчастіше файли WAV зберігають звук в початковому вигляді без стиснення. WAV займає більше пам'яті, ніж популярні MP3 чи AAC. WAV складається із двох частин: заголовку та інформації. Опісля послідовності символів інформації починаються дані аудіофайлу, але наступні 4 байти показують довжину інформації. Знаючи структуру WAV, береться результати blockAlign (blockAlign показує, скільки байтів описує один семпл). У наявності двох каналів (стерео) перша половина blockAlign показуватиме перший потік, але друга половина – другий потік. Метод ISB надає сховати секретну дані у молодших бітах каналів.

1.8 Опис засобів розроблювання програмного застосунку

У створенні програмного застосунку стего-системи задля передавання та захищення схованих інформації використані ці засоби задля програмування мовою C#, мов Microsoft Visual Studio 2022 і Windows Forms (рис.1.27). Мова C# проста в використанні і водночас повноцінна мова програмування, що надає багато засобів задля структурування та підтримки великих програм і рішень. Вона краще поза C/C++ обробляє помилки, та, будучи мовою високого рівня, вміщує вбудовані типи інформації високого рівня, ці мов гнучкі масиви, списки

та словники, ефективна реалізація котрих на мові C потребує значних витрат часу. Разом з цим задля розширення функціональності треба застосовувати готові бібліотеки, котрі отримуються напряму у середовища розроблювання через вбудований в Visual Studio 2022 менеджер пакетів NuGet Package Manager.

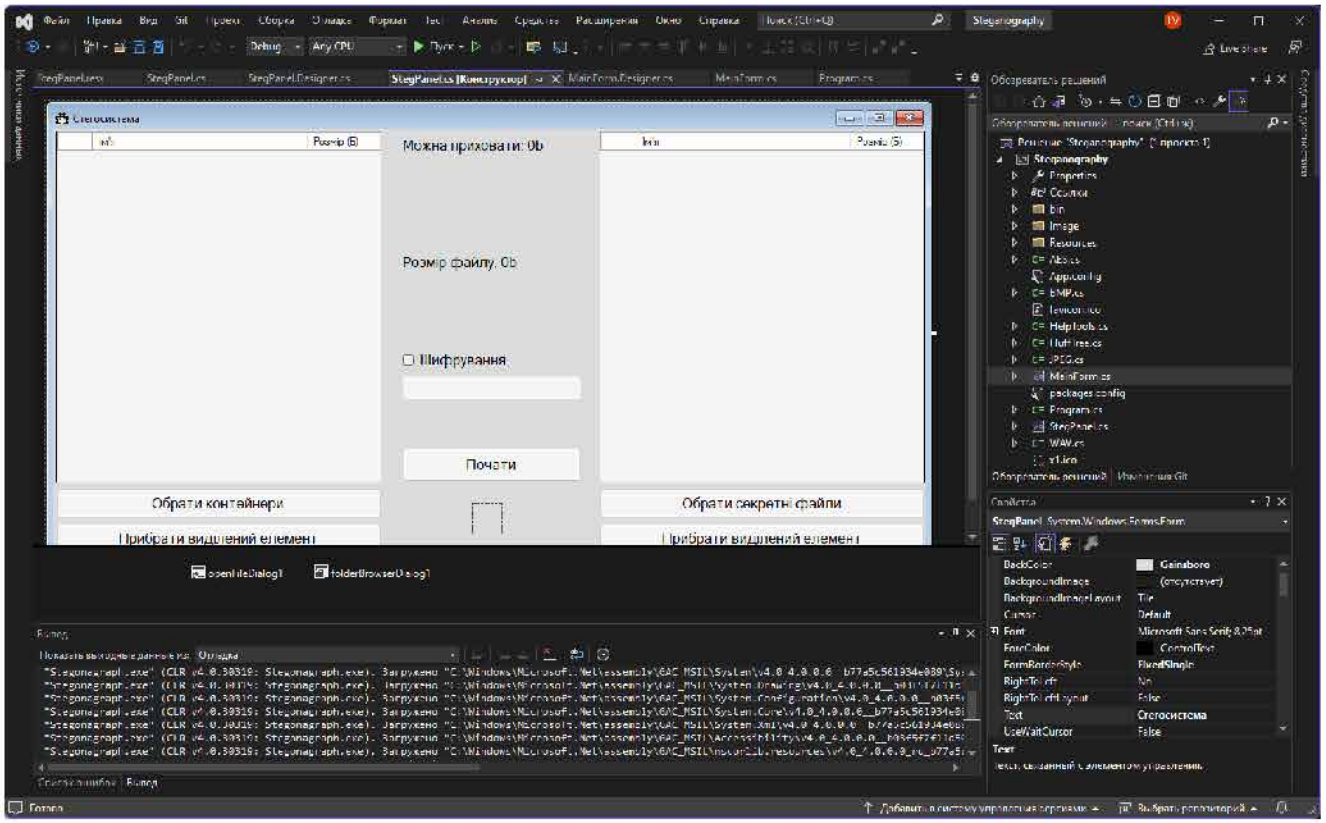


Рисунок 1.27. Середовище розроблювання Microsoft Visual Studio 2022

Мова програмування C# надає розбивати програми на модулі, що потім можуть бути використані у інших програмах. C# поставляється із великою кількістю стандартних бібліотек, котрі треба застосовувати, мов основу задля нових програм чи мов приклади у вивченні мови. Стандартні модулі надають засоби задля роботи із файлами, системними викликами, мережними із'єднаннями та навіть інтерфейсами щодо різних графічних бібліотек. C# надає писати зручні задля читання програми завдяки загальноприйнятим узгодженням щодо написання коду і назв полів різних типів.

Синтаксис C# близький щодо C++ та Java. Мова вміщує строгу статичну типізацію, підтримує поліморфізм, наслідування, перевантаження операторів, інкапсуляцію, закриття методів, вказівники на функції і члени класів, атрибути, події, властивості, делегати, винятки, коментарі в форматі XML. Перейнявши

багато чого з своїх попередників (мов C++, Delphi та Smalltalk) – C#, спираючись на практику їхнього впровадження, виключає деякі моделі, що зарекомендували себе мов проблематичні у розробці програмних систем, наприклад, множинне успадкування класів (на відміну з C++)

Windows Forms надає розробляти інтелектуальні клієнти. Інтелектуальний клієнт – це програма із повнофункціональним графічним інтерфейсом, просте у розгортанні та оновленні, здатне працювати у наявності чи відсутності підключення щодо Інтернету та використовує більш безпечний доступ щодо ресурсів на локальному комп'ютері у порівнянні із традиційними застосунками Windows. Windows Forms – це технологія інтелектуальних клієнтів задля .NET Framework. Вона являє собою набір керованих бібліотек, що спрощують реалізація стандартних завдань, таких мов читання із файлової системи та запис у неї. У використанні середовища розроблювання, мов Visual Studio, треба створювати інтелектуальні клієнтські програми Windows Forms, котрі відображають відомості, запитують введення з користувачів та обмінюються даними із віддаленими комп'ютерами по мережі. В Windows Forms, форма – це візуальна поверхня, на якій виводиться інформація задля користувача. Зазвичай застосунок Windows Forms будується шляхом приміщення складових керування на форму та написання коду задля реагування на дії користувача, ці мов клацання миші чи натискання клавіш.

Елемент керування – це окремий елемент призначеного задля користувача інтерфейсу, призначений задля відображення чи введення інформації. У виконанні користувачем якої-небудь дії із формою чи із одним із складових керування створюється подія. Застосунок реагує на ці події поза поміччю коду та обробляє події у їх виникненні. Windows Forms включає широкий набір складових керування, котрі треба додавати на форми: текстові поля, кнопки, списки, що розкриваються, перемикачі і навіть веб-сторінки. Коли існуючий елемент керування не задовольняє потребам, у Windows Forms треба створювати власні складові керування. Щодо складу Windows Forms входять багатофункціональні складові призначені задля користувача інтерфейсу, що

дозволяють відтворювати можливості таких складних застосунків, мов Microsoft Office. Використовуючи необхідні складові керування, треба створювати панелі інструментів та меню, що містять текст та малюнки, і інші складові керування, ці мов текстові поля та поля зі списками.

Поза поміччю Visual Studio треба легко створювати застосунки Windows Forms. Досить виділити елемент керування курсором та помістити його у потрібне місце на формі. Задля подолання труднощів, пов'язаних із вирівнюванням складових керування, конструктор надає ці додаткові складові, мов лінії сітки та лінії прив'язки. Поза поміччю Visual Studio чи компіляції із командного рядка, треба застосовувати складові керування задля створення складних макетів форм поза менший час. В багатьох застосунках потрібно відображати дані із бази інформації, XML-файлу, веб-служби XML чи іншого джерела інформації. Windows Forms надає гнучкий елемент керування задля відображення таких табличних інформації у традиційному форматі рядків та стовпців так, що кожен фрагмент інформації займає свою власну клітинку. Поза його поміччю треба, налаштувати зовнішній вигляд окремих осередків, зафіксувати рядки та стовпці на своєму місці, але разом з цим забезпечити відображення складних складових керування всередині осередків. Поза поміччю Windows Forms треба легко створювати складові керування із прив'язкою щодо інформації. Створювати складові керування із прив'язкою щодо інформації треба шляхом перетягування об'єктів із допоміжного вікна у форми проекту.

Застосунки, складені мовою на C#, працюють на платформі .NET Framework, вбудованому компоненті Windows, котрий включає віртуальну систему середовища реалізація під назвою Common Language Runtime (CLR), але разом з цим набір інтегрованих класів та бібліотек. CLR з Microsoft є реалізацією міжнародного стандарту Common Language Infrastructure (CLI), котрий є середовищем створення, запуску і розроблювання, де мови програмування працюють разом без бар'єрів. Вихідний код C# компілюється у проміжну мову (IL), що надає специфікації CLI. Код та ресурси IL, включаючи растрові фото і рядки, зберігаються на диску мов виконуваний файл (зазвичай із розширенням

.exe чи .dll). Колекція вміщує маніфест, котрий надає дані про типи, версію колекції, вимоги безпеки, мову і регіональні налаштування.

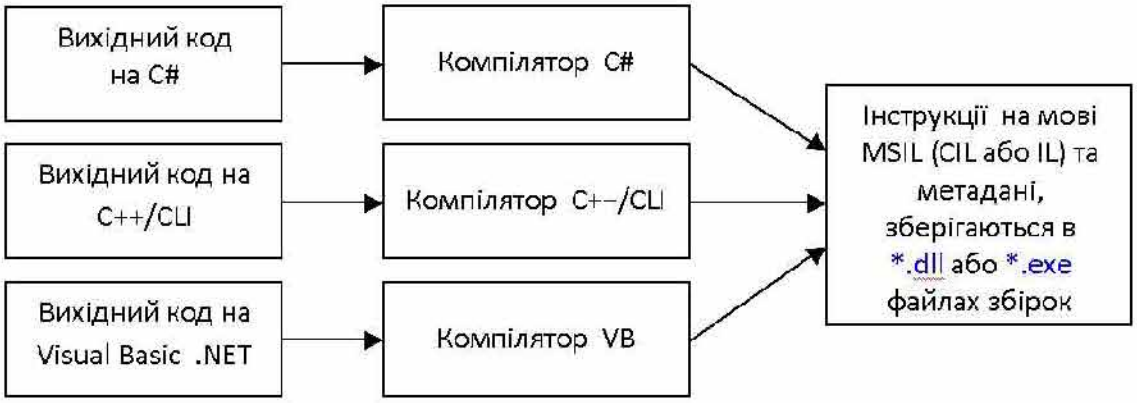


Рисунок 1.28. Процес перетворення вихідного коду у код на мові MSIL і утворення файлу збірки

Коли реалізується програма C#, CLR завантажує колекцію і реалізує різні операції залежно з даних, що зберігається у маніфесті. Коли всі вимоги безпеки виконано, середовище CLR реалізує JIT-компіляцію із IL-коду у машинний код. CLR разом з цим реалізує інші операції, ці мов автоматичне збирання сміття, обробка винятків та керування ресурсами. На рис.1.28 показано взаємозв'язки поміж програмними файлами C#, бібліотеками класів .NET Framework, збірками і середовищем CLR під час компіляції і реалізація.

1.9 Розробка структури стего-системи задля передавання та захищення схованих інформації

У розробці структури стего-системи передбачені можливі дії користувача і реакція на них із формуванням відповідних результатів. У необхідності сховати окремі файли користувач вміщує спочатку обрати файли-контейнери, в котрих треба сховати секретну дані. Бокс проходить процес аналізу формату, аби визначити, формат (bmp/png wav та jpg) та чи достатнього він розміру задля приховання відповідних по розміру секретних файлів. Користувач вміщує разом з цим ввести код задля впровадження під час кодування секретного файлу. Процес приховання даних вміщує починатись опісля реалізація аналізу формату бокса і вибору методів кодування.

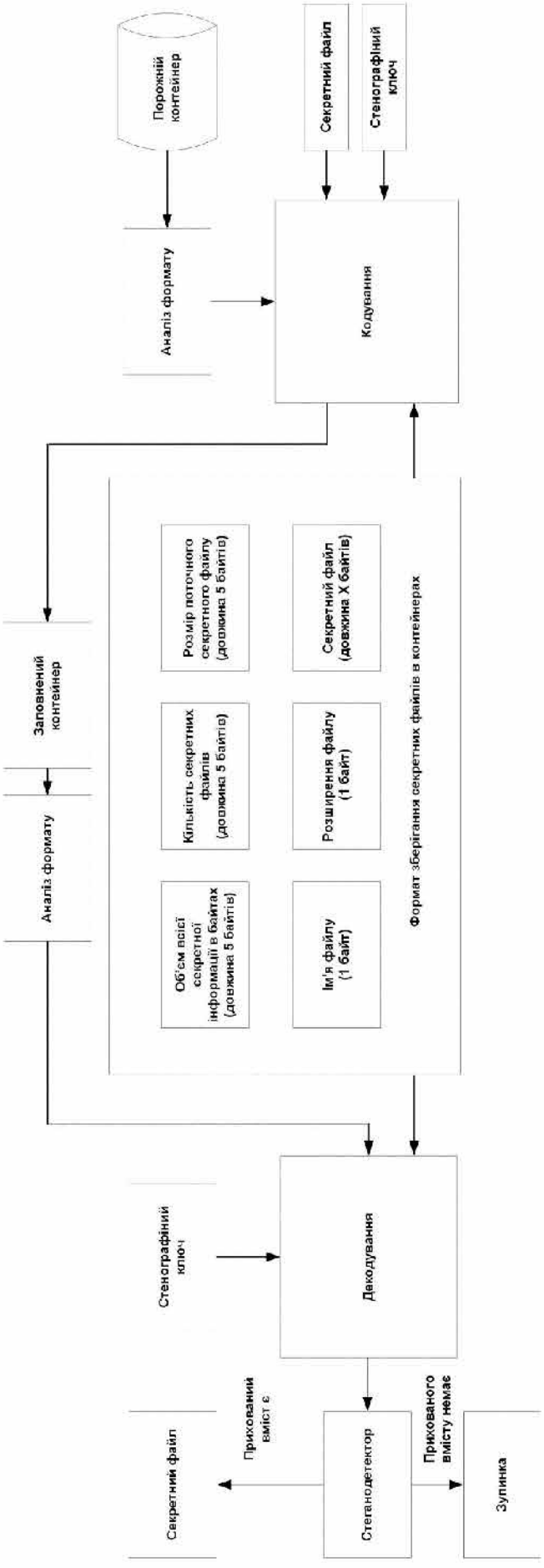


Рисунок 1.29. Структура стего-системи задля передавання та захищення схованих

інформації

Секретний файл та код мають перетворюватися на певний формат, котрий буде передаватися одержувачу. Опісля кодування буде створюватись заповнений бокс, що вміщує приховану секретну дані. У виконанні зворотної операції бокс вміщує зазнавати процес дешифрація, використовуючи відповідні моделі дешифрація. Коли код правильний, секретний файл вміщує декодуватися і витягуватися із бокса. Потім бокс вміщує перевірятись на наявність схованої даних. Коли стеганодетектор виявить приховані дані, із контейнеру вміщує почати витягуватись секретний файл. Коли стеганодетектор не виявить жодних схованих інформації, це буде означати, що бокс порожній.

Окрім дешифрація секретного файлу, користувач розроблюваної стеганосистеми буде мати можливість застосовувати той самий чи інший код задля зашифровування секретної даних. Коли секретна інформація шифрована, вона вміщує розшифровуватися безпосередньо опісля дешифрація. На рис. 1.29 показано послідовність дій, що виконуються в програмі задля приховання і вилучення секретної даних із бокса, використовуючи моделі кодування і дешифрація.

Формат зберігання секретних файлів у контейнерах					
Об'єм всієї секретної даних у байтах (довжина 5 байтів)	Число секретних файлів (довжина 5 байтів)	Розмірність поточного секретного файлу (довжина 5 байтів)	Ім'я файлу (1 байт)	Розширення файлу (1 байт)	Секретний файл (довжина X байтів)
1	2	3	4	5	6

Рисунок 1.30. Структура формату зберігання в контейнерах секретних файлів

Секретні файли будуть зберігатися в контейнері структурою, що відображена на рис. 1.30, у якій вся інформація буде кодуватися так:

– перші п'ять байтів зберігають обсяг байтів всіх секретних файлів;

– наступні п'ять байтів зберігають обсяг конкретного секретного файлу;

– наступні п'ять байтів зберігають розмірність поточного секретного файлу;

– наступний 1 байт зберігає ім'я файлу. Коли він більше 255 біт, то беруться

перші 255 біт;

– наступний 1 байт зберігає розширення конкретного секретного файлу. Воно спроможне бути будь-якого типу;

– в наступних бітах зберігається секретний файл, розмірність якого був визначений.

1.10 Розробка структури діаграми класів стего-системи задля передавання та захищення схованих інформації

В програмній реалізації стего-системи мовою C# в середовищі розроблювання Visual Studio передбачено основний об'єкт MainForm, що є головним вікном програми. В головному вікні користувач вводить код зашифровування та обирає дію, яку треба виконати (сховати чи виявити вміст). Об'єкт StegPanel надає поза обробку приховання і вилучення даних із контейнеру. Об'єкт StegPanel вміщує зв'язки із кількома іншими класами. Він спроможне взаємодіяти із класом BMP/PNG, аби сховати чи отримати дані фото в форматі BMP чи PNG.

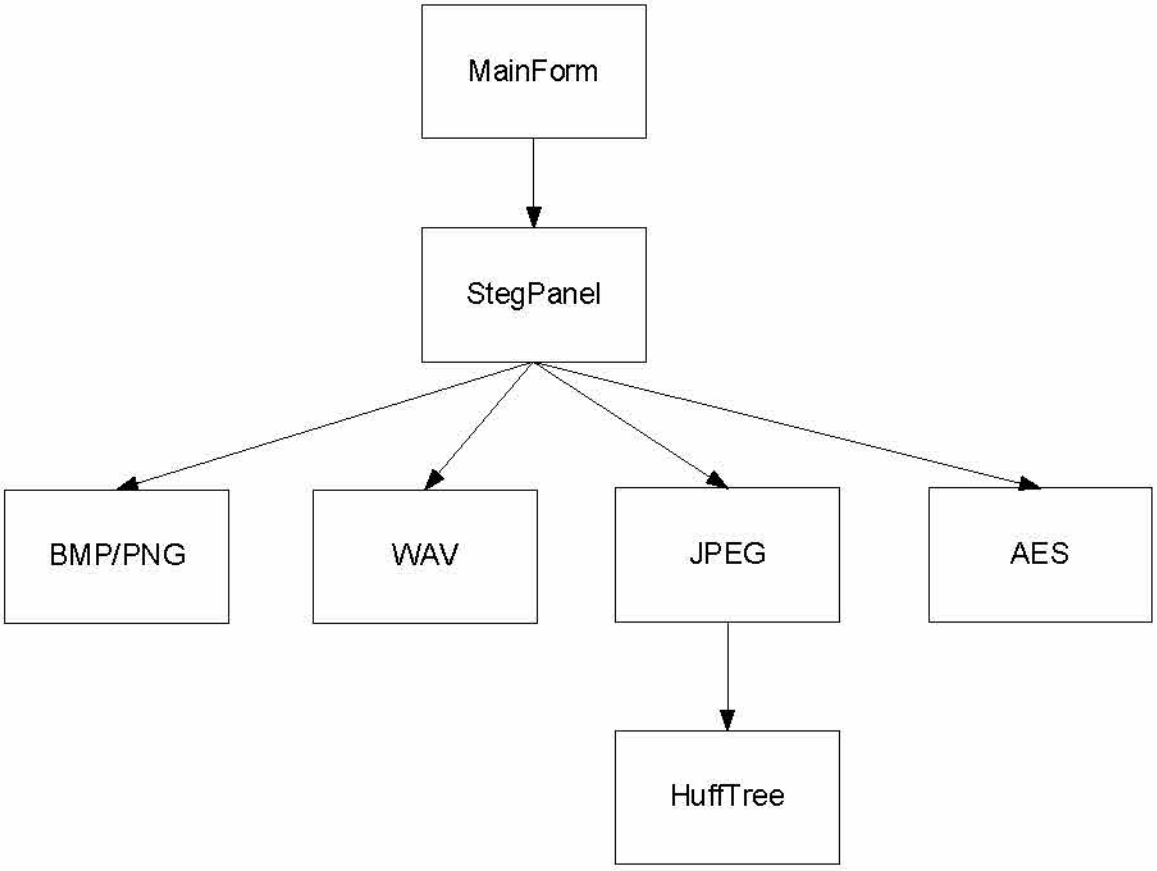


Рисунок 1.31. Діаграма взаємодії класів в програмній реалізації стего-системи

Разом з цим він вміщує зв'язок із класом JPG, котрий використовує стеганографію в форматі JPG із деревом Хафмана (HuffTree) задля приховання і вилучення даних. Об'єкт StegPanel разом з цим використовує об'єкт WAV задля приховання чи вилучення даних із аудіофайлів в форматі WAV. Окрім того, об'єкт StegPanel вміщує можливість взаємодіяти із класом A-E-S, що надає шифрувати і дешифрувати приховану дані із використанням відповідного алгоритму зашифровування. На рис. 1.31 показано діаграму взаємодії поміж класами в програмній реалізації стего-системи, що реалізує разом з цим зашифровування. Кожен об'єкт надає певним функціям та забезпечує можливість взаємодії із іншими класами задля реалізація необхідних операцій.

1.10.1 Реалізація структури об'єкту MainForm в застосунку

Головна форма проекту MainForm вміщує складові керування, ці мов кнопки (button1, pbHide, pbUnhide), мітку SteganographyKeyLabel і текстове поле textBoxStegKey. Об'єкт MainForm визначено відповідними полями і методами. Поля button1, components, pbHide, pbUnhide, SteganographyKeyLabel і textBoxStegKey пов'язані із елементами керування форми і іншими компонентами, необхідними задля функціонування проекту (рис. 1.32).

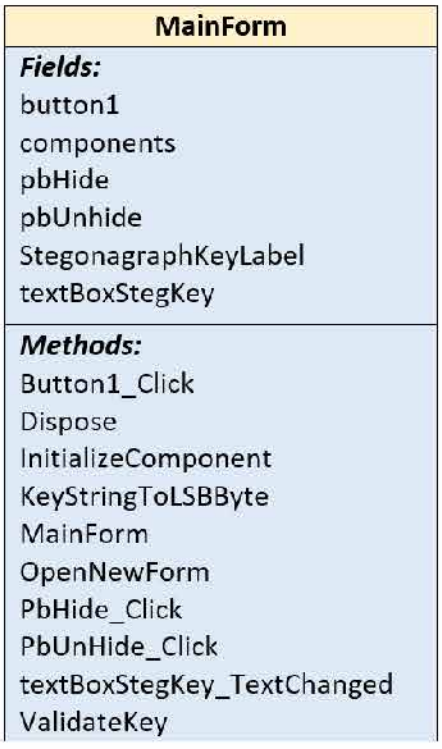


Рисунок 1.32. Складові об'єкту MainForm в застосунку

Моделі об’єкту MainForm Button1_Click, Dispose, InitializeComponent, KeyStringToLSBByte, MainForm, OpenNewForm, PbHide_Click, PbUnHide_Click, textBoxStegKey_TextChanged і ValidateKey відповідають поза обробку подій, ініціалізацію компонентів, перетворення ключа на байти, створення нових форм і валідацію ключа.

1.10.2 Реалізація структури об’єкту StegPanel в застосунку

Дочірня форма проекту StegPanel вміщує складові панелі керування процесом приховання і відновлення даних в проекті. Об’єкт StegPanel вміщує поля і моделі задля керування стеганографічним процесом.

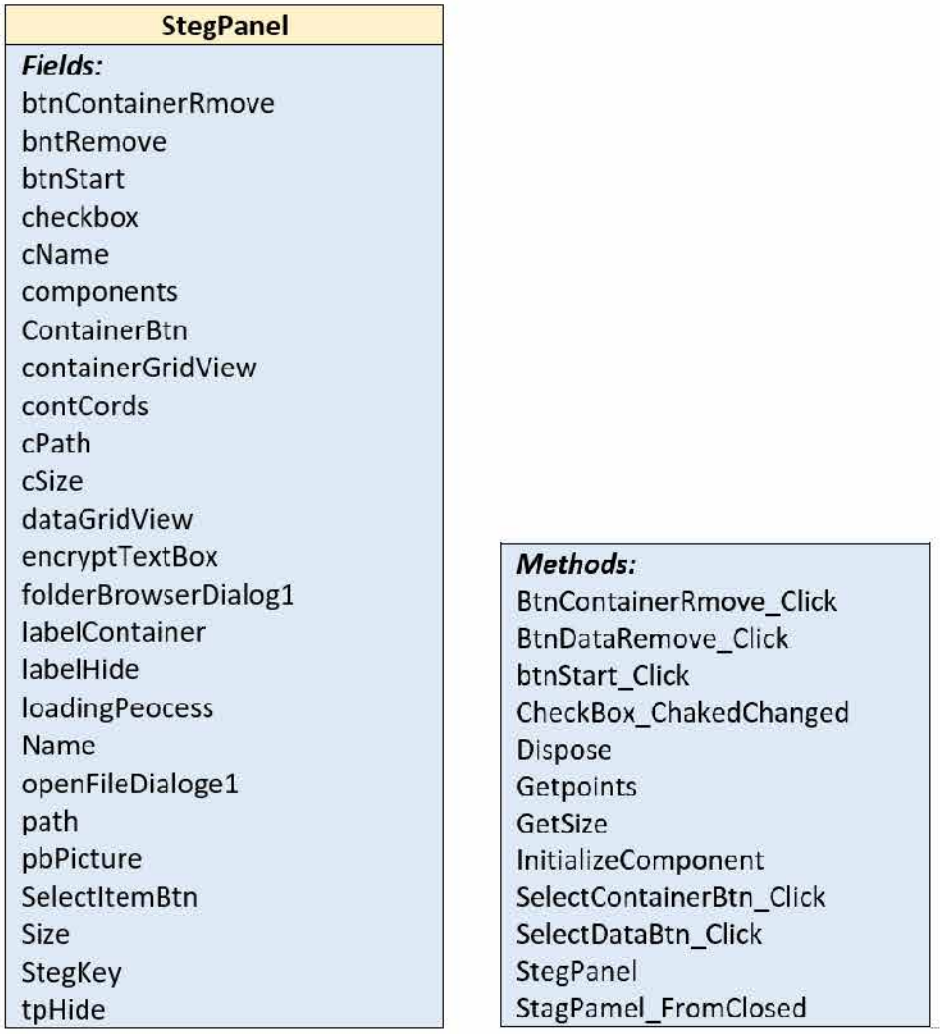


Рисунок 1.33. Складові об’єкту StegPanel в застосунку

Поля об’єкту StegPanel, але саме btnContainerRemove, btnRemove, btnStart, checkBox, cName, components, ContainerBtn, containerGridView, contCoords, cPath, cSize, dataGridView, encryptTextBox, folderBrowserDialog1, labelContainer

pbPicture, SelectItemBtn, Size, StegKey представляють складові керування, дані і стани, пов'язані із панеллю стегографії. Наприклад, btnContainerRemove і btnRemove пов’язані із кнопками задля видалення бокса і видалення секретних файлів, btnStart – кнопка задля запуску стеганографічного процесу, checkBox – прапорець задля активації зашифровування поза алгоритмом AES.

Моделі об’єкту StegPanel, зокрема BtnContainerRemove_Click, BtnDataRemove_Click, btnStart_Click, CheckBox_ChakedChanged, Dispose, GetPoints, GetSize, InitializeComponent, SelectContainerBtn_Click, SelectDataBtn_Click, Steg відповідають поза обробку подій, ініціалізацію компонентів, вибір складових, налаштування, запуск процесу стегографії і обробку закриття форми панелі (рис.1.33).

1.10.3 Реалізація структури об’єкту BMP/PNG в застосунку

В розроблюваному застосунку об’єкт BMP/PNG надає функціональність, пов'язану із обробкою фото в форматі BMP і PNG, пов'язану із цифровою стеганографією. Об’єкт BMP/PNG, застосовується у проекті задля роботи із зображеннями в форматі BMP, впровадження стеганографічних методів задля приховання і вилучення даних із цих фото. Він вміщує декілька методів задля кодування і дешифрація даних в зображеннях BMP і PNG. Об’єкт BMP/PNG вміщує моделі bmpDecode, bmpEncode, ReadFromBitmap і WriteToBitmap. Метод bmpDecode надає поза дешифрація схованої даних із фото BMP і PNG із використанням способу останнього біта (ISB). Метод bmpEncode надає поза кодування даних і її приховання в зображенні BMP і PNG. Метод ReadFromBitmap реалізує читання інформації із фото BMP задля подальшої обробки чи аналізу. Метод WriteToBitmap реалізує запис інформації фото BMP і PNG (рис. 1.34).

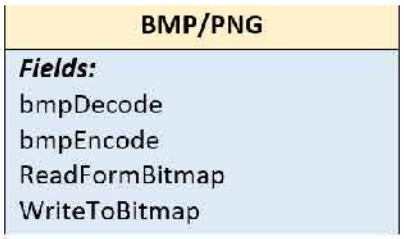


Рисунок 1.34. Складові об’єкту BMP/PNG в застосунку

1.10.4 Реалізація структури об’єкту WAV в застосунку

В розроблюваному застосунку об’єкт WAV надає функціональність, пов'язану із обробкою звукових файлів в форматі WAVE. Він вміщує кілька полів і методів задля роботи із даними звукових файлів. Поля об’єкту WAV включають wavFile, котрий представляє собою об'єкт чи шлях щодо звукового WAVE-файлу. Об’єкт WAV вміщує властивості AudioInfoCount, BitsPerSample, BlockAlignBytes, NumberOfChannels і StartPos. Ці властивості надають дані про характеристики звукового файлу WAV, ці мов число аудіофрагментів, число біт на вибірку, розмірність блоку у байтах, число каналів і початкова позиція в файлі. Об’єкт WAV вміщує моделі Wav, WavDecode і WavEncode. Метод Wav служить конструктором об’єкту, що ініціалізує об'єкт WAV. Метод WavDecode надає поза дешифрація схованої даних із звукового файлу WAV. Метод WavEncode надає поза кодування. Об’єкт WAV застосовується у проекті задля роботи зі звуковими файлами в форматі WAV, впровадження способу останнього біта, приховання і вилучення даних із цих файлів.

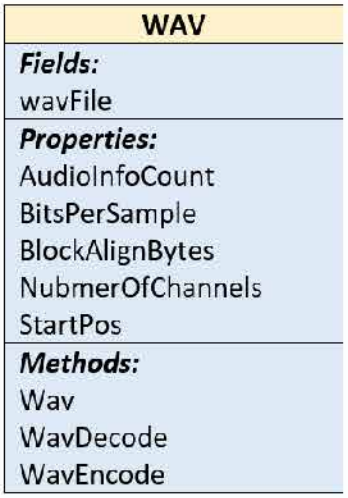


Рисунок 1.35. Складові об’єкту WAV в застосунку

1.10.5 Реалізація структури об’єкту JPG в застосунку

В розроблюваному застосунку об’єкт JPG надає функціональність, пов'язану із обробкою фото в форматі JPG. Він вміщує декілька полів і методів задля роботи із даними фото в форматі JPG. Об’єкт JPG вміщує поля arrayJpeg, compCount, DHT_AC і DHT_DC. Поле arrayJpeg представляє собою масив

інформації, що містять фото в форматі JPG. Поле compCount наводить число масштабів фото. Поля DHT_AC та DHT_DC відносяться щодо таблиць Хафмана задля кодування значень масштабів яскравості і кольоровості у JPG. Об'єкт JPG вміщує властивості info і secretfiles. Властивість info надає дані про JPG-фото, зокрема його розмірність. Властивість secretfiles представляє бітовий потік секретних файлів. Метод GetInfo призначений задля отримання даних про те, скільки спроможне зберігати секретних файлів JPG-фото. Метод JPG є конструктором об'єкту, що ініціалізує об'єкт JPG. Метод JpegCompressor застосовується задля стиснення фото і конвертування в базовий режим формату JPG. Метод jpegDecode надає поза дешифрація схованої даних із JPG-фото. Метод jpegEncode надає поза кодування даних і її приховання в зображенні JPG. Метод jpegProcessing реалізує обробку фото JPG (рис. 1.36). Об'єкт JPG застосовується у проекті задля роботи із зображеннями в форматі JPG, впровадження стеганографічних методів задля приховання і вилучення даних із цих фото.

Рисунок 1.36. Складові об'єкту JPG в застосунку

1.10.6 Реалізація структури об’єкту HuffTree в застосунку

В розроблюваному застосунку об’єкт HuffTree представляє функціональність, пов'язану із побудовою і використанням дерев Хафмана. Дерева Хафмана можуть бути використані задля кодування і дешифрація секретної даних в процесі стегографії JPG файлів. Об’єкт HuffTree вміщує властивості Code і Val, але разом з цим метод HuffTree. Властивість Code представляє кодове результати дерева Хафмана. Властивість Val зберігає результати, що надає коду дерева Хафмана. Метод HuffTree служить задля побудови дерева Хафмана на основі вхідних інформації (рис. 1.37).

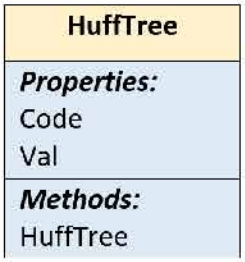


Рисунок 1.37. Складові об’єкту HuffTree в застосунку

1.10.7 Реалізація структури об’єкту A-E-S в застосунку

В розроблюваному застосунку об’єкт A-E-S надає функціональність, пов'язану із алгоритмом зашифровування A-E-S (Advanced Encryption Standard). Він вміщує кілька полів і методів, пов'язаних із процесом зашифровування і дешифрування інформації. Об’єкт A-E-S вміщує поля GMatrix, invGMatrix, invSBox, Rcon, roundKey і SBox. Ці поля містять матриці і таблиці, що використовуються у алгоритмі A-E-S задля перетворення інформації. Моделі об’єкту A-E-S включають конструктор A-E-S, котрий реалізує ініціалізацію об'єкта об’єкту A-E-S. Моделі Decrypt і Encrypt відповідають поза дешифрування і зашифровування інформації відповідно, із використанням перевантажень задля різних вхідних параметрів. Моделі DecryptForOneCycle і EncryptForOneCycle виконують один цикл зашифровування чи розшифрування. Метод gmul реалізує галуа-перетворення (галуа-множення) задля двох чисел. Моделі InvRotCell, InvSubBytes, KeySchedule, MatrixMultiplication, RotCell і SubBytes відповідають поза різні перетворення, що використовуються у алгоритмі A-E-S (рис. 1.38).

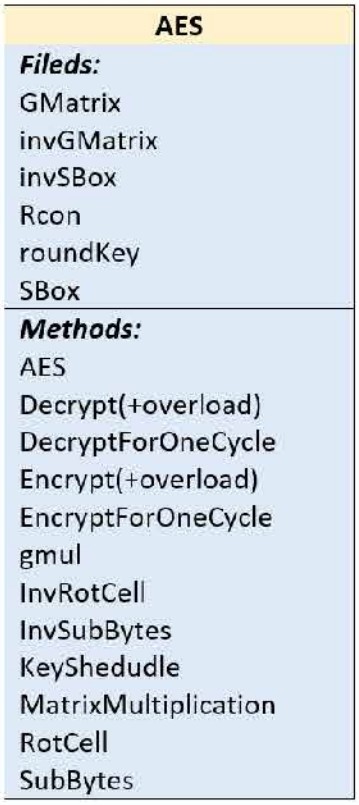


Рисунок 1.38. Складові об'єкту A-E-S в застосунку

1.11 Розробка візуального інтерфейсу стего-системи

Розроблена стегосистема призначена задля передавання та захищення схованих інформації. Програма дозволить приховувати будь-котрий тип формату секретних файлів в контейнерах форматів PNG, BMP, JPG та WAV.

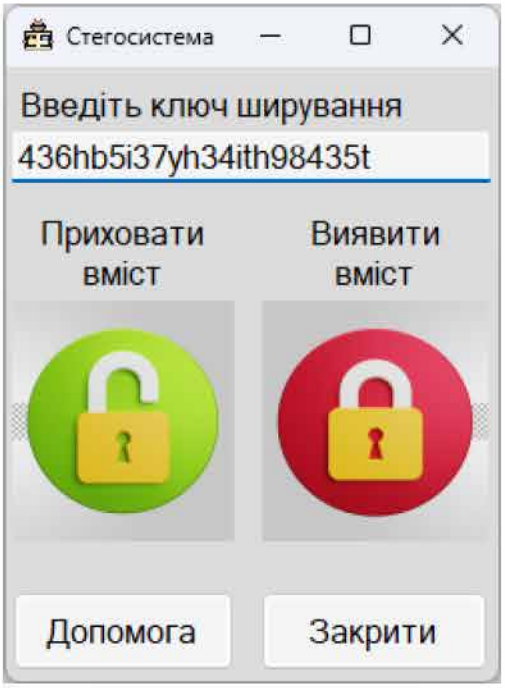


Рисунок 1.39. Головне вікно програмного застосунку стего-системи

Приховувані дані треба додатково захистити поза поміччю алгоритму зашифровування AES128. Програмний застосунок не обмежує розмірність приховуваних файлів, але під час роботи із великими файлами задля стегографії спроможне знадобитися багато часу. Код програми мовою C# наведений в додатку АЛЕ. Візуальний інтерфейс головного вікна програмного застосунку показано на рис.1.39.

В інтерфейсі головного вікна передбачено ці складові візуального інтерфейсу користувача:

– активна область “Сховати вміст” – у натисканні на зображенні відкритого замку відкривається дочірнє вікно, у якому реалізується приховання обраних секретних файлів в контейнерах і відповідне кодування;

– активна область “Виявити вміст” – у натисканні на зображенні закритого замку відкривається дочірнє вікно, у якому реалізується відновлення секретних файлів із контейнерів і відповідне дешифрація;

– поле вводу “Введіть код зашифровування” – передбачає створення стеганографічного ключу перед початком кодування;

– кнопка “Допомога” – опісля натискання відкривається pdf-файл із інструкцією задля користувача.

– кнопка “Закрити” – опісля натискання програмний застосунок буде закрито і вивантажено із пам’яті.

У натисненні активних областей “Сховати вміст” і “Виявити вміст” буде відкрито дочірнє вікно керування процесом стегографії (рис.1.40).

В інтерфейсі дочірнього вікна задля керування процесом стегографії передбачені наступні можливості відповідно щодо рис.1.40:

1. Обрати контейнери, в котрих будуть приховуватись секретні файли;

2. Обрати файли, котрі треба сховати;

3. В таблиці показано всі обрані контейнери, котрі використовуватимуться задля стегографії;

4. В таблиці показано файли, котрі будуть приховані в контейнерах;

5. Кнопка надає видалити обрані складові із таблиці контейнерів;

6. Кнопка надає видалити обрані складові із таблиці секретних файлів;

7. Кнопка надає запустити процес стегографії;

8. Відображується розмірність бокса та, відповідно, кількості даних (у байтах), яку треба сховати;

9. Відображується розмірність приховуваного файлу (у байтах);

10. Прапорець надає шифрувати секретні файли поза алгоритмом A-E-S із використанням додаткового паролю зашифровування.

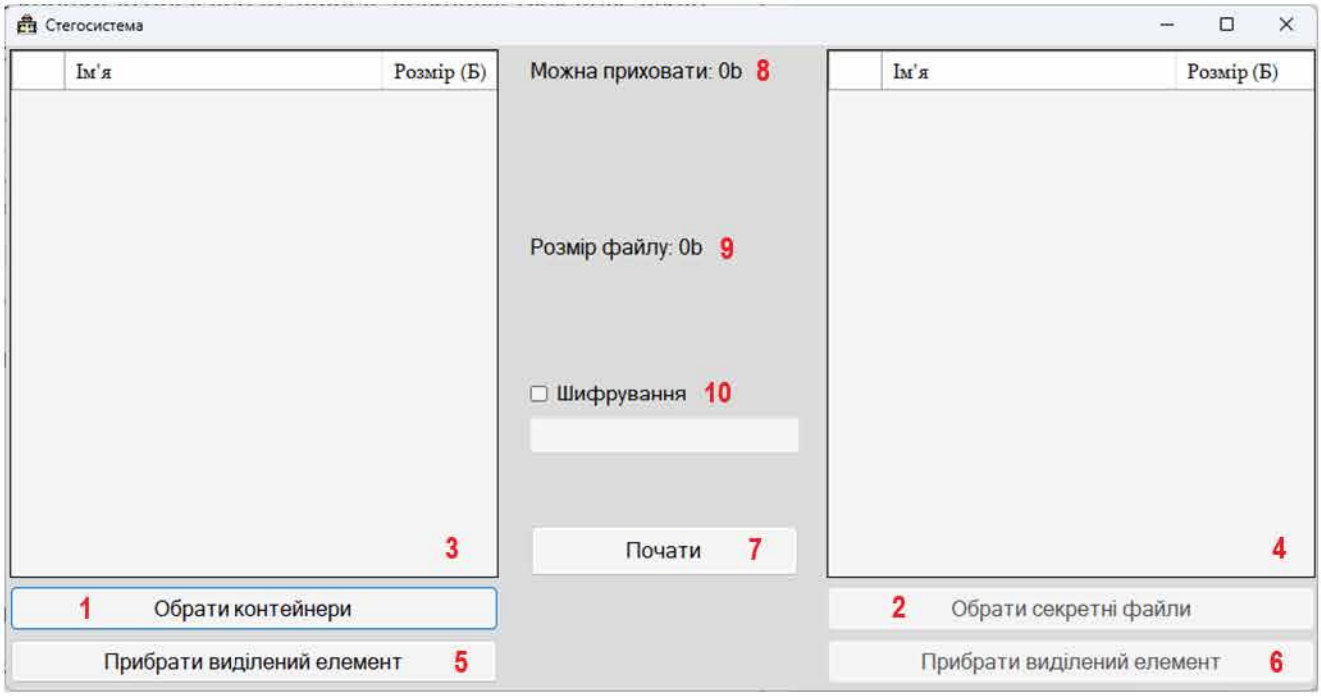


Рисунок 1.40. Дочірнє вікно програмного застосунку стего-системи і панелі керування процесом приховання і відновлення даних

Аби обрати бокс, треба вибрати перший стовпець в таблиці (3). В таблиці (3) другий стовпець відображує назву бокса, але третій стовпець відображує максимальний розмірність секретного файлу, котрий бокс спроможне сховати (рис.1.41).

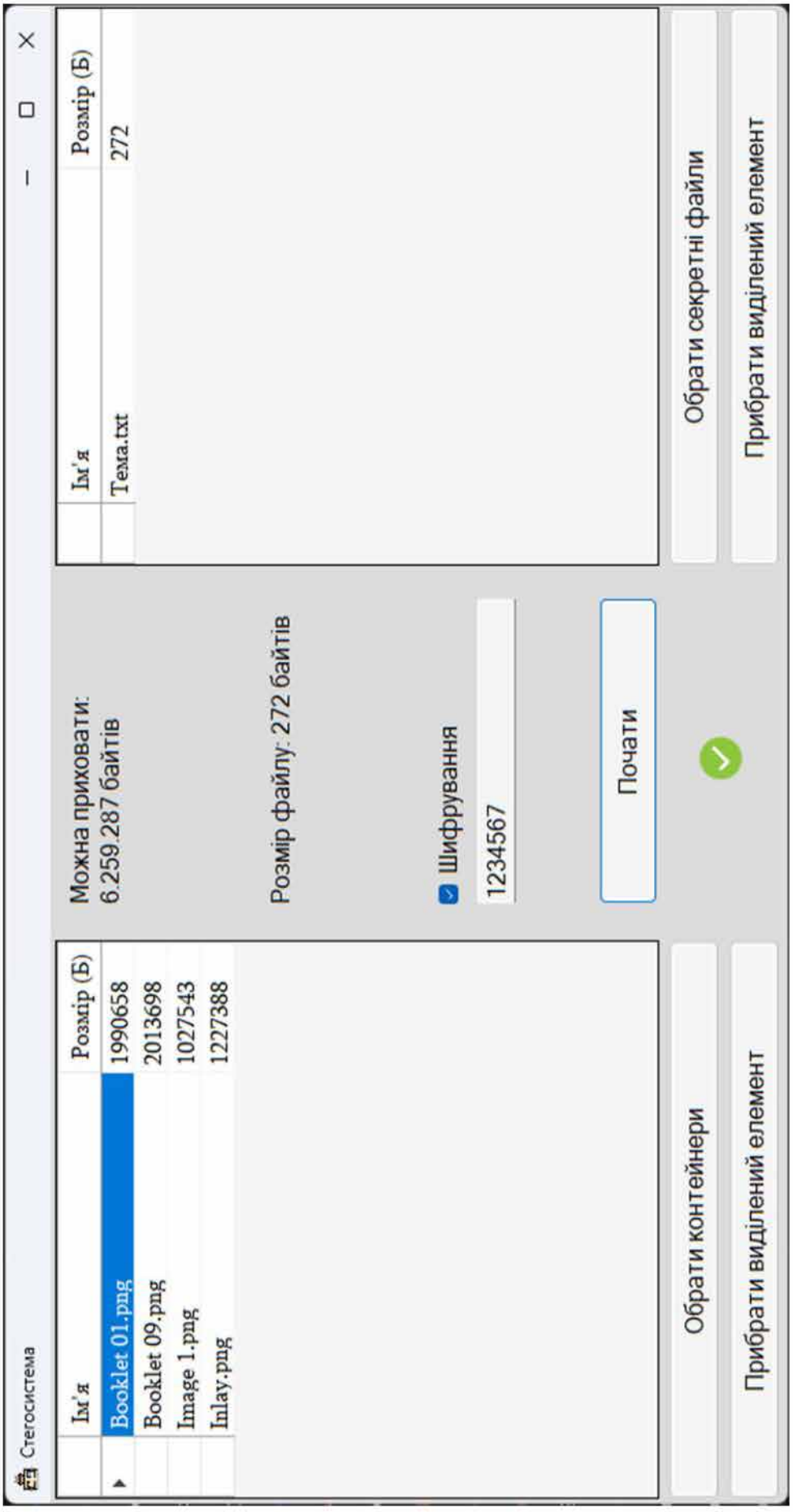


Рисунок 1.41. Процес приховання даних в PNG-контейнері

2 ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

У даному дипломному проекті виконано програмну реалізацію стего-системи задля передавання та захищення схованих інформації. ІЗ її поміччю треба значною мірою полегшити та оптимізувати роботу із охорони і обліку робочого часу співробітників на підприємстві та скоротити витрати на утримання робочого персоналу. Впровадження створеного програмного забезпечення дозволить перейти на новий етап впровадження високонадійних систем безпеки, у тому числі та задля режимних об'єктів.

Ефективність кожного програмного продукту визначається його якістю і ефективністю процесу розроблювання. Якість ПП визначається наступними складовими: із точки зору користувача; із позиції впровадження ресурсів; реалізація вимог щодо програмного забезпечення. Оцінка якості програмного продукту із точки зору користувача визначається необхідним на стадії функціонування розміром оперативної пам'яті ЕОТ, витратами машинного часу, пропускною спроможністю каналів передавання інформації. Оцінка якості програмного продукту включає визначення трудомісткості та вартості його створення.

2.2 Визначення трудомісткості розроблювання програмного забезпечення

Тривалість розроблювання програмного продукту залежить з його обсягу, трудомісткості розроблювання, кваліфікації виконавців, але разом з цим планових термінів, визначених умовами ринку. Методом структурної аналогії по відповідних каталогах аналогів програмного забезпечення визначається обсяг програмних засобів, в тисячах умовних машинних команд програми аналога.

Таблиця 2.1 Каталог аналогів

Найменування ПП	Обсяг функції ПП – V_o, усл. машинних командах.
1. ПП автоматизованих розрахунків	1300 – 8600
2. Комплексні системи ведення БД	950 – 7430
3. ПП введення даних	1060 – 5750

В таблиці 2.1 представлені аналоги програмного забезпечення, функції котрих, в більшому чи меншому ступені, реалізує розроблений програмний продукт. Задля нашого варіанта виділено сірим кольором.

Вибравши аналог ПП, що вміщує Vo у умовних машинних командах, трудомісткості визначати на основі табл.2.2

Таблиця 2.2

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

На підставі отриманого результати, по довіднику, визначається укрупнена норма часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розроблювання ПП, тобто у умовах комп'ютера, Кк=О,7÷О,8): I = 229 х О,8 = 183,2 (люд/годин).

Трудомісткість програмного продукту визначається по кожному етапу розроблювання окремо на підставі трудомісткості аналога із урахуванням складності

розроблювання, ступеня новизни та ступеня впровадження у розробці стандартних модулів на підставі формул:

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

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

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

Задля розрахунку необхідні наступні коефіцієнти:

Li – питома вага і-го етапу розроблювання (див. табл. 2.2.);

Кн – поправочний множник, що враховує ступінь новизни (див. табл. 2.3.);

Кт – поправочний множник, що враховує ступінь впровадження у розробці типових програм (див. табл. 2.4.).

Таблиця 2.2 Результати питомих масштабів трудомісткості стадії у загальній трудомісткості розроблювання ПП

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

Задля нашого варіанта виділено сірим кольором.

Таблиця 2.3 Результати поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	Ступінь новизни	Результати $К_н$
АЛЕ	Принципово нові ПП	1,75 – 1,2
Б	ПП – розвиток визначеного параметричного ряду	1,О – О,8
У	ПП маючий аналог	О,7

Задля нашого варіанта виділено сірим кольором.

Таблиця 2.4 Результати коефіцієнта ступеня впровадження у розробці типових програм

Ступінь охоплення реалізованих функцій розроблювального ПП типовими програмами, %	Результати $К_т$
60 та вище	О,6
40-60	О,7
20-40	О,8
Щодо 20	О,9

Задля нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$Т_{тз}=I*L_1*К_н= 183,2 *О,12*О,7= 15,39 \text{ (люд/годин)}$$

Трудомісткість розроблювання технічного проекту

$$Т_{тп}=I*L_2*К_н =183.2 *О,11*О,7 = 17,42 \text{ (люд/годин)}$$

Трудомісткість розроблювання робочого проекту

$$Т_{рп}=I*L_3*К_н*К_т =183,2 *О,61*О,7*О,7=54,76 \text{ (люд/годин)}$$

Задля подальших розрахунків визначили число папера, витраченого на кожен етап: технічне завдання $N_{тз}$= 2 (стор), розробка ТП $N_{тп}$=18 (стор), розробка робочого проекту $N_{рп}$=25(стор), пояснювальна записка відповідно $N_{пз}$= 25 (стор)Розрахунок зведений в таблицю 2.5

Таблиця 2.5 Розрахунок трудомісткості ПП

Найменування етапів	Розрахунок, годин.		
1	2	3	4
1.ТЗ	$T_{Ртз}$=15,39	$T_{кк}$=О,7*$N_{тз}$= О,7*2=1,4	$T_{нк}$=О,15*$N_{тз}$=О,15*2= О,30
2.Розробка ТП	$T_{Ртп}$=14,12	$T_{кк}$=О,7*$N_{тп}$=О,7*18= 12,6	$T_{нк}$=О,15*$N_{тп}$=О,15*18 =2,7
3.Розробка РП	$T_{Ррп}$=54,76	$T_{кк}$=О,7*$N_{рп}$=О,7*25= 17,5	$T_{нк}$=О,15*$N_{рп}$=О,15*25 =3.8
4.Розробка ПЗ	$T_{пз}$=1,5* *$N_{пз}$=1,5*2 5=37,5	$T_{кк}$=О,7*$N_{тз}$=О,7*25= 17,57	$T_{нк}$=О,15*$N_{пз}$=О,15*25 =3,8
Усього, у т.ч.:	ΣТ=181,4		
- на розробку	$ΣT_{р}$=121,8		
- контроль керівника		$ΣT_{кк}$=49	
-нормоконтроль			$ΣT_{нк}$=10,6

2.3 Розрахунок ціни програмного продукту

В цьому розділі задля визначення ціни розраховуємо основну заробітну плату виконавців, матеріальні витрати, вартість машино – години та витрати на розробку ПО. Розрахунок основної заробітної плати виконавців приведений в таблиці 4.6. Відповідно щодо статті 8 «Закону про Державний бюджет України на 2022» встановлено мінімальну заробітну плату в місячному розмірі із 1 січня 2022 року - 6500 гривень; мінімальну погодинну тарифну ставку – 39.26 грн.

Таблиця 2.6 Розрахунок основної заробітної плати виконавців

Найменування робіт	Трудомісткість робіт, години	Погодинна тарифна ставка, грн.	Розрахунок, грн.
1.Розробка ПП	121,8	39.26	4781,87
2.Контроль керівника	49	50.00	2450
3.Нормоконт-роль	10,6	50.00	530
Усього	-	-	ΣЗо= 7761,87

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо у таблицю 2.7

Таблиця 2.7 Розрахунок матеріальних витрат на розробку ПО

Найменування матеріальних витрат	Тип, модель	Число	Ціна одиниці, грн.	Вартість, грн.
Папір	Лист А4	80	2,00	160,00
				$В_{м\,та}$=160,00
Транспортно– заготівельні витрати (10%)				$Втр\_з = 0{,}1 \times Вм1$= 1,60
Усього				$В_м$=$В_{мі}$+$В_{тр\ із}$= 176,00

На підставі отриманих інформації по окремих статтях витрат складена калькуляція планової собівартості у цілому ПП поза формою, приведеною у таблиці 2.8.

Таблиця 2.8. Розрахунок статей витрат планової собівартості

Стаття витрат	Результати, грн.	Формула розрахунку
1. Матеріали	176,00	$В_м$(див. табл. 4.7)
2. Основна заробітна плата	7761,87	$З_о$(див. табл. 4.6)
3.Додаткова заробітна плата	1164,28	$Зд = 0{,}15 \times Зо$=7761,87
4.Відрахування щодо єдиного фонду соціального внеску	8926,15	$Вє.с.в. = 0{,}22 \times (Зо + Зд)$=О,22х(7761,87+ 1164,28)
5. Накладні витрати	2328,56	$Внак. = 0{,}3 \times Зо$=О,3 х7761,87
6. Повна собівартість	20356,86	$С_{пов}$= $В_м$+$З_о$+ $З_д$+Вє.с.у. + Внак.

Розмірність прибутку, що включається у ціну, визначаємо по наступній формулі:

$П=(С_{пов}*Р)/100=20356,86*10/100=2035,68$ (2.4)

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

Оптова ціна (кошторисна вартість) визначається по формулі:

$Ц_о =С_{пов}+П =$ 20356,86+ 2035,68 =22392,54 (2.5)

Податок на додану вартість визначаємо по наступній формулі:

$ПДВ=О.2*Ц_о=О,2*22392,54=4478,51$ (2.6)

Виходячи із отриманих інформації, ціна реалізації розробленого програмного продукту на основі наступної формули, становитиме:

$Ц_р=Ц_о+ПДВ=22392,54+4478,51=26871,05$ (2.7)

3 РОЗДІЛ ОХОРОНИ ПРАЦІ І ТЕХНІКИ БЕЗПЕКИ

Безпека праці, спрямована на створення небезпечних та нешкідливих умов праці. На сучасному етапі розвитку виробництва вона набуває все більше важливого результати.

Вирішення завдань охорони праці базується на досягненнях ергономіки, наукової організації праці, технічної естетики, гігієни і фізіології праці, психофізіології. Крім того, успіх охорони праці визначається темпами впровадження передової техніки, підвищення рівня механізації та автоматизації виробничих процесів, удосконаленням технології і організації виробництв

Безпека праці на підприємстві спроможне бути на належному рівні тільки тоді, коли всебічно надає вимогам трудового законодавства, державним стандартам України, норм та правил, розроблених задля збереження здоров'я працюючих. Важливе місце у цьому належить виконанню організаційних вимог із охорони праці, але разом з цим трудовій і виробничій дисципліні працюючих.

Дипломний проектом передбачена програмна реалізація стего-системи задля передавання та захищення схованих інформації. Реалізація даної роботи проводилося поза поміччю персонального комп'ютера. В зв'язку із цим треба проаналізувати фактори ризику у роботі із сучасним персональним комп'ютером.

3.1 Аналіз небезпечних і шкідливих чинників, що впливають на працівника

Основними факторами шкідливого впливу ПК на організм людини є ці:

1. Електромагнітні поля;
2. Електромагнітні випромінювання;
3. Розгортка фото на моніторі;
4. Мелькання фото на екрані;
5. Тривала нерухомість пози оператора.

Сукупний вплив на людину всіх шкідливих факторів знижує загальний біоенергетичний потенціал та опірність організму, знижує імунітет, викликає м'язову атрофію та застої у органах. Наслідки порушення норм безпеки у роботі

поза ПК можуть викликати професійні захворювання чи призвести щодо нещасного разі і травмування працівника.

3.2 Розробка заходів із охорони праці

Зменшити вплив перерахованих факторів ризику та зберегти здоров'я людині, що постійно використовує у роботі ПК, надає дотримання всіх заходів та засобів, передбачених охороною праці.

3.2.1 Мікроклімат робочої зони працівників, вентиляція

В виробничих приміщеннях на робочих місцях із ВДТ мають забезпечуватись оптимальні результати параметрів мікроклімату: температури, відносної вологості та рухливості повітря (ГОСТ 12.1.005-88, СН 4088-86).

Рівні позитивних та негативних іонів у повітрі приміщень із ВДТ повинні задовольняти санітарно-гігієнічним нормам № 2152-80

3.2.2 Освітлення робочого місця, шум, вібрація

Штучне освітлення у приміщеннях із робочими місцями, обладнаними ЕОМ та ПЕОМ, вміщує здійснюватись системою загального рівномірного освітлення. Результати освітленості на поверхні робочого столу у зоні розміщення документів вміщує становити 300 - 500 лк.

Мов джерело світла у штучному освітленні застосовуються переважно люмінесцентні лампи.

Результати освітленості на поверхні робочого столу у зоні розміщення документів вміщує становити 300 - 500 лк.

Система загального освітлення вміщує становити суцільні чи переривчасті лінії світильників, розташовані збоку з робочих місць (переважно зліва), паралельно лінії зору працюючих Впровадження світильників без розсіювачів і екрануючих грат заборонено.

Рівні звукового тиску у октавних смугах частот мають відповідати вимогам СН 3223-85, ГОСТ 12.1.003-83, ГР 2411-81.

Задля забезпечення допустимих рівнів шуму на робочих місцях слід застосовувати засоби звукопоглинання, У виконанні робіт із ЕОМ в виробничих приміщеннях результати характеристик вібрації на робочих місцях не повинні

перевищувати допустимі згідно СН 3044-84, ГОСТ 12.1.012-90. У розумовій праці, що вимагає зосередженості припустимий рівень шуму становить 50дБ

3.2.3 Організація робочого місця користувача ПК

- Важливо, аби офісний працівник сидячи поза комп'ютером знаходився поза добре освітленим робочим столом. Найчастіше саме погане освітлення робочого місця надає більш згубний задля зору вплив, ніж сам факт перебування поза комп'ютером.
- Робочі столи слід розміщувати цим способом, аби монітори були орієнтовані бічною стороною щодо світлових прорізів, аби природне світло падало переважно ліворуч.
- У розміщенні робочих місць відстань поміж робочими столами повинна бути не менше 2,О м, але відстань поміж бічними поверхнями відеомоніторів - не менше 1,2 м.
- Конструкція робочого столу повинна забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання.
- Конструкція робочого стільця чи крісла повинна забезпечувати підтримку раціональної робочої пози працівника.
- Клавіатуру слід розташовувати на поверхні столу на відстані 100..300 мм з краю, зверненого щодо користувача, чи на спеціальній поверхні, відокремленій з основної стільниці.
- Екран відеомонітора повинен знаходитися з очей користувача на відстані 600-700 мм, але не ближче 500мм.

Рисунок 3.1. Організація робочого місця оператора ПК

Безпека праці у роботі поза комп'ютером передбачає, що тривалість безперервної роботи поза комп'ютером без регламентованої перерви не повинна перевищувати 2 години.

Не рекомендується працювати поза комп'ютером більше 6 годин поза зміну. Рекомендується робити перерви у роботі поза ПК тривалістю 10 хвилин через кожні 50 хвилин роботи. Під час регламентованих перерв доцільно виконувати комплекси вправ.

У нерегламентованій роботі підвищеної інтенсивності можливі головні болі, нервові зриви і інше.

3.3 Пожежна безпека

Протипожежна безпека на підприємстві – невіддільна частина організації робочого простору та процесів згідно із нормами чинного законодавства. Зокрема, цю сферу регламентують Правила пожежної безпеки у Україні, затверджені наказом Міністерства внутрішніх справ України, зі змінами, котрі періодично вносяться відповідними наказами.

Попри обладнання будівель будь-якими типами установок пожежогасіння, пожежної сигналізації чи внутрішніми пожежними кранами, офісні приміщення разом з цим мають бути забезпечені первинними засобами пожежогасіння.

Щодо первинних засобів пожежогасіння належать: вогнегасники, кошма (покривало із негорючого теплоізоляційного полотна), ящики із піском, бочки із водою, пожежні відра, багри, ломи, сокири тощо. Найбільш зручними задля впровадження є вогнегасники.

Відповідальними поза своєчасне і повне оснащення об'єктів засобами пожежогасіння, забезпечення їх технічного обслуговування, навчання працівників правил користування ними є роботодавець і керівники структурних підрозділів.

Відповідальні особи зобов'язуються розробити протипожежний режим та інструкції відповідно щодо вимог, викладених у нормативних актах на зазначених їм об'єктах.

Встановлений режим включає порядки із описом місць спеціального

призначення і правила їх користування і утримання, наприклад:

- евакуаційних шляхів,
- так званих «курилок»,
- місць складування продукції і сировини,
- стоянки транспорту.

Важливою складовою протипожежного режиму на будь-якому об’єкті є розробка та впровадження порядку дій у виникненні пожежі. Неодмінно вміщує бути план евакуації, описано, мов повинні відключатися електроустановки, що та у якій послідовності треба робити співробітникам.

ВИСНОВКИ

В дипломному проекті виконано програмну реалізацію стего-системи задля передавання та захищення схованих інформації і проведено дослідження стеганографічних методів у галузі захищення даних.

У ході роботи розглянуто сучасні комп'ютерної стегографії, вивчено її особливості і основні поняття. Проведено моделювання і проектування програмної системи, що використовує моделі стегографії задля захищення інформації в мультимедійних файлах (зображеннях чи звуках). Описано структуру системи, вхідні дані і впровадження способу ISB задля стеганографічного приховання даних. Метод ISB гарантує, що розмірність файлу не зміниться навіть опісля кодування. Це дозволило приховувати в файлах-контейнерах набагато більший об'єм секретної даних в порівнянні із іншими алгоритмами.

У розробці програмного застосунку стего-системи використовувалось середовище розроблювання Microsoft Visual Studio та .NET Framework і мова програмування C#. Реалізовано структуру класів і їх функціональність.

Програмний застосунок стего-системи задля передавання та захищення схованих інформації, розроблений в даній роботі, продемонстрував добрі результати у застосуванні способу заміни найменш значущого біта задля приховання даних в зображеннях різного формату. Однак, було виявлено, що впровадження способу заміни найменш значущого біта є вразливим щодо стеганоалітичних атак, котрі можуть виявити наявність та природу схованої даних.

Задля забезпечення вищого рівня захищення схованої даних в зображенні застосовано криптографічні моделі. Перед поміщенням схованої даних в фото сповіщення шифрувалося із використанням криптоалгоритму A-E-S. Отримані результати сприятимуть покращенню захищення конфіденційної даних і забезпеченню безпеки у передавання і зберіганні інформації в цифровому середовищі.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Конахович Г.Ф. Комп'ютерна стеганографічна обробка й аналіз мультимедійних даних: навчальна література. Центр навч. літ., 2018. – 560с.
2. Конахович Г.Ф. Пузиренко А.Ю. Комп'ютерна стеганографія. Теорія та практика. // Київ: МК-Пресс., 2006 р. – 288 с.
3. Кузнецов О.О. Стеганографія: навчальний посібник – Х. : Вид. ХНЕУ, 2011. – 232 с.
4. Хорошко В.О. Основи комп'ютерної стеганографії: навч. посібн. для студентів і аспірантів – Вінниця : ВДТУ, 2003. – 143 с.
5. Коноваленко І.В. Програмування мовою C# 6.0: навч. посіб. – Тернопіль, ТНТУ-2016 – 229с.
6. Кузьміних В.О. Управління версіями програмних засобів проекту: Навчальний посібник – КПІ ім. Ігоря Сікорського, 2023.
7. Цибульник С. О., Барандич К. С. Технології розроблення програмного забезпечення: Навчальний посібник – КПІ ім. Ігоря Сікорського, 2022.
8. К.Т. Кузьма, В.О. Поздєєв. Основи об'єктно-орієнтованого програмування мовою C#: Навчальний посібник – МНУ, 2022.
9. Color Image Quantization: A Short Review and an Application with Artificial Bee Colony Algorithm - IOS Press. Home - IOS Press. [Електронний ресурс]: https://content.iospress.com/articles/informatica/inf25-3-08
10. Албахарі Б. C# 7.0. Довідник. Повний опис мови / Б. Албахарі, А. Албахари., 2018.
11. C# Documentation: [Електронний ресурс]: https://learn.microsoft.com/uk-ua/dotnet/csharp/ (англійською мовою).
12. The Art of Hiding Information – Johannes Trithemius' Steganography. [Електронний ресурс]: http://scihi.org/johannes-trithemius-steganography/
13. Alan Siper Roger Farley and Craig Lombardo. The Rise of Steganography. Seidenberg School of CSIS Homepage | Pace University New York. [Електронний ресурс]: URL: https://csis.pace.edu/~ctappert/srd2005/d1.pdf.

ДОДАТОК А

Вміст файлу StegPanel.cs з кодом мовою C# проекту стегосистеми

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace Stegonagraph
{
   public partial class StegPanel : Form
   {
      List<Point> contCords = new List<Point>();
      Boolean tpHide;
      Process loadingPeocess;
      byte[] stegKey;
      public StegPanel(Boolean hideUnhide,byte[] stegKey)
      {
         InitializeComponent();
         containerGridView.Columns[1].SortMode = DataGridViewColumnSortMode.NotSortable;
         this.stegKey = new byte[stegKey.Length];
         for (int i = 0; i < stegKey.Length; i++)
            this.stegKey[i] = stegKey[i];
                dataGridView.Font = new Font("Sylfaen", 12);
         containerGridView.Font = new Font("Sylfaen", 12);
         if (!hideUnhide)
         {
            dataGridView.Enabled = false;
            SelectItemBtn.Enabled = false;
            btnRemove.Enabled = false;
         }
         tpHide = hideUnhide;
         contCords.Add(new Point(20, 20 - new TrackBar().Height));
               }
      private void SelectContainerBtn_Click(object sender, EventArgs e)
      {
         UInt64 info = 0;
         DialogResult result = openFileDialog1.ShowDialog();
         if (result == DialogResult.OK)
         {
            try {
loadingPeocess = Process.Start(System.IO.Directory.GetCurrentDirectory() + "//Resources//WaitForm.exe");
            }
            catch (Exception err) {}
            foreach (String file in openFileDialog1.FileNames)
            {
               FileInfo fileInfo = new FileInfo(file);
               Bitmap bp;
               String Name = fileInfo.Name;
               Name = Name.Substring(0, Name.LastIndexOf('.'));
Name = Name.Length > 255 ? Name.Substring(0, 255) + fileInfo.Extension : Name + fileInfo.Extension;
               Boolean bl = false;
               for (int i = 0; i < containerGridView.Rows.Count; i++)
               {
                  if (Name == containerGridView.Rows[i].Cells[0].Value.ToString())
                  {
                     bl = true;
                     MessageBox.Show("Файл вже існує!");
                  }
```

```
            }
            if (bl)
                continue;
            switch (fileInfo.Extension.ToLower())
            {
                case ".bmp":
                case ".png":
                    bp = new Bitmap(fileInfo.FullName);
                    pbPicture.Image = Image.FromFile("Image/true.png");
                    info = 0;
                    for (int i = 0; i < bp.Width * bp.Height; i++)
                        info += (ulong)(3 * stegKey[i % stegKey.Length]);
                                containerGridView.Rows.Add(fileInfo.Name, info / 8, fileInfo.FullName);
                    break;
                case ".wav":
                    WAV wavInfo = new WAV(fileInfo.FullName);
                    pbPicture.Image = Image.FromFile("Image/true.png");
                    info = 0;
                    for (UInt64 i = 0; i < wavInfo.AudioInfoCount/(ulong)wavInfo.BlockAlignBytes; i++)
                        info += (ulong)stegKey[i % (ulong)stegKey.Length];
                    if (wavInfo.NumberOfChannels == 1)
                        containerGridView.Rows.Add(fileInfo.Name, info / 8, fileInfo.FullName);
                    else
                        containerGridView.Rows.Add(fileInfo.Name, (info * 2) / 8, fileInfo.FullName);
                    break;
                case ".jpg":
                case ".jpeg":
                    JPEG jpeg = new JPEG(fileInfo.FullName, dataGridView.Enabled ? true : false);
                    jpeg.GetInfo(stegKey);
                    pbPicture.Image = Image.FromFile("Image/true.png");
                    containerGridView.Rows.Add(fileInfo.Name, jpeg.info, fileInfo.FullName);
                    break;
                default:
                    pbPicture.Image = Image.FromFile("Image/false.png");
                    break;
            }
        }
        try{loadingPeocess.Kill();}
        catch (Exception err) { }
        containerGridView.ClearSelection();
        containerGridView.CurrentCell = null;
    }
    labelContainer.Text = "Можна приховати: " + GetPoints(GetSize(containerGridView)) + " байтів";
}
private void SelectDataBtn_Click(object sender, EventArgs e)
{
    DialogResult result = openFileDialog1.ShowDialog();
    if (result == DialogResult.OK)
    {
        foreach (String file in openFileDialog1.FileNames)
        {
            FileInfo fileInfo = new FileInfo(file);
            byte[] btArray = File.ReadAllBytes(file);
            String Name = fileInfo.Name;
            Name = Name.Substring(0, Name.LastIndexOf('.'));
Name = Name.Length > 255 ? Name.Substring(0, 255) + fileInfo.Extension : Name + fileInfo.Extension;
            Boolean bl = false;
            for (int i = 0; i < dataGridView.Rows.Count; i++)
            {
                if (Name == dataGridView.Rows[i].Cells[0].Value.ToString())
                {
                    bl = true;
                    MessageBox.Show("Файл вже існує!");
                }
            }
            if (bl)
                continue;
            String name = fileInfo.Name.Substring(0, fileInfo.Name.LastIndexOf('.'));
```

```
dataGridView.Rows.Add(fileInfo.Name, (dataGridView.Rows.Count == 0 ? 10 : 0) +  1 (name.Length>255?510:
 name.Length*2) + 1 + fileInfo.Extension.Substring(1).Length + 5 + btArray.Length   , fileInfo.FullName);
            }
            dataGridView.ClearSelection();
            dataGridView.CurrentCell = null;
        }
        labelHide.Text = "Розмір файлу: " + GetPoints(GetSize(dataGridView)) + " байтів";
    }
    private UInt64 GetSize(DataGridView dgv)
    {
        UInt64 byteCount = 0;
        for (int i = 0; i < dgv.Rows.Count; i++)
            byteCount += Convert.ToUInt64(dgv.Rows[i].Cells[1].Value.ToString());
        return byteCount;
    }
    private String GetPoints(UInt64 num)
    {
        String str = num.ToString();
        String endStr = "";
        while (str.Length > 3)
        {
            endStr = "." + str.Substring(str.Length - 3, 3) + endStr;
            str = str.Substring(0, str.Length - 3);
        }
        return str + endStr;
    }
    private void btnStart_Click(object sender, EventArgs e)
    {
        List<Byte> hideInfo = new List<Byte>();
        if (containerGridView.Rows.Count == 0)
        {
            MessageBox.Show("Вставте контейнер!");
            return;
        }
        if (dataGridView.Rows.Count == 0 && dataGridView.Enabled)
        {
            MessageBox.Show("Вставте приховану інформацію!");
            return;
        }
        if (checkBox.Checked && encryptTextBox.Text.Length == 0)
        {
            MessageBox.Show("Вставте пароль шифрування!");
            return;
        }
        if(GetSize(containerGridView) < GetSize(dataGridView))
        {
            MessageBox.Show("Розмір секретного файлу більше, ніж розмір контейнеру!");
            return;
        }
        DialogResult result = folderBrowserDialog1.ShowDialog();
        String savePath = "";
        if (result != DialogResult.OK) {
            return;
        }
        savePath = folderBrowserDialog1.SelectedPath + "\\";
        try
        {
loadingPeocess = Process.Start(System.IO.Directory.GetCurrentDirectory() + "//Resources//WaitForm.exe");
        }
        catch (Exception err) { }
        if (tpHide)
        {
            String[] pathArray = new String[dataGridView.Rows.Count];
            UInt64 hideFileCount = (UInt64)GetSize(dataGridView);
            for (int i = 0; i < 5; i++) {
                hideInfo.Add((byte)(hideFileCount >> ((4 - i) * 8)));
            }
```

```
            hideFileCount = (UInt64)pathArray.Length;
            for (int i = 0; i < 5; i++)
               hideInfo.Add((byte)(hideFileCount >> ((4 - i) * 8)));
            for (int i = 0; i < pathArray.Length; i++)
               pathArray[i] = dataGridView.Rows[i].Cells[2].Value.ToString();
            for (int i = 0; i < pathArray.Length; i++)
            {
               FileInfo fileInfo = new FileInfo(pathArray[i]);
               String fileName = fileInfo.Name.Substring(0, fileInfo.Name.IndexOf("."));
               if (fileName.Length > 255)
                  fileName = fileName.Substring(0, 255);
               String fileExt = fileInfo.Extension.Substring(1);
               hideInfo.Add((byte)fileName.Length);
               for (int j = 0; j < fileName.Length; j++)
               {
                  Encoding.ASCII.GetBytes(fileName[j].ToString());
                  hideInfo.Add((Encoding.Unicode.GetBytes(fileName[j].ToString()))[0]);
                  hideInfo.Add((Encoding.Unicode.GetBytes(fileName[j].ToString()))[1]);
               }
               hideInfo.Add((byte)fileExt.Length);
               for (int j = 0; j < fileExt.Length; j++)
                  hideInfo.Add((byte)fileExt[j]);
               byte[] btHideArray = File.ReadAllBytes(pathArray[i]);
               UInt64 secretLength = (UInt64)btHideArray.Length;
               for (int j = 0; j < 5; j++)
                  hideInfo.Add((byte)(secretLength >> ((4 - j) * 8)));
               for (long j = 0; j < btHideArray.Length; j++)
                  hideInfo.Add(btHideArray[j]);
            }
            if (checkBox.Checked)
            {
               AES cryptPattern = new AES(encryptTextBox.Text);
               hideInfo = cryptPattern.Encrypt(hideInfo.ToArray()).OfType<byte>().ToList();
            }
            pathArray = new String[containerGridView.Rows.Count];
            for (int i = 0; i < pathArray.Length; i++)
               pathArray[i] = containerGridView.Rows[i].Cells[2].Value.ToString();
            for (int i = 0; i < containerGridView.Rows.Count; i++)
            {
               if (hideInfo.Count == 0)
                  break;
int hideCount = hideInfo.Count > Convert.ToInt32(containerGridView.Rows[i].Cells[1].Value) ?
Convert.ToInt32(containerGridView.Rows[i].Cells[1].Value) : hideInfo.Count;
               FileInfo fileInfo = new FileInfo(pathArray[i]);
               switch (fileInfo.Extension.ToLower())
               {
                  case ".png":
                  case ".bmp":
BMP.bmpEncode(hideInfo.GetRange(0, hideCount), savePath + fileInfo.Name,
new Bitmap(pathArray[i]), stegKey);
                     break;
                  case ".wav":
                     WAV shablon = new WAV(containerGridView.Rows[i].Cells[2].Value.ToString());
                     shablon.WavEncode(hideInfo.GetRange(0, hideCount), savePath + fileInfo.Name, stegKey);
                     break;
                  case ".jpg":
                  case ".jpeg":
                     JPEG jpeg = new JPEG(pathArray[i],true);
    jpeg.jpegEncode(hideInfo.GetRange(0, hideCount).ToArray(), savePath + fileInfo.Name, stegKey);
                     break;
               }
               hideInfo.RemoveRange(0, hideCount);
            }
            try { loadingPeocess.Kill(); }
            catch (Exception err) { }
            MessageBox.Show("Інформація прихована!");

         }
```

```
        else
        {
            String[] pathArray = new String[containerGridView.Rows.Count];
            List<byte> findInfo = new List<byte>();
            for (int i = 0; i < pathArray.Length; i++)
                pathArray[i] = containerGridView.Rows[i].Cells[2].Value.ToString();
            for (int i = 0; i < containerGridView.Rows.Count; i++)
            {
                FileInfo fileInfo = new FileInfo(pathArray[i]);
                switch (fileInfo.Extension.ToLower())
                {
                    case ".png":
                    case ".bmp":
                        findInfo.AddRange(BMP.bmpDecode(new Bitmap(pathArray[i]),
Convert.ToInt32(containerGridView.Rows[i].Cells[1].Value),stegKey));
                        break;
                    case ".wav":
                        WAV shablon = new WAV(containerGridView.Rows[i].Cells[2].Value.ToString());
findInfo.AddRange(shablon.WavDecode(Convert.ToInt32(containerGridView.Rows[i].Cells[1].Value), stegKey));
                        break;
                    case ".jpg":
                    case ".jpeg":
                        JPEG jpeg = new JPEG(pathArray[i], false);
                        jpeg.jpegDecode(stegKey);
                        findInfo.AddRange(jpeg.secretFiles);
                        break;
                }
            }
            try
            {
                UInt64 fileQanakByte, fileCount;
                List<byte> qanak = new List<byte>();
                qanak = findInfo.GetRange(0, 16);
                if (checkBox.Checked)
                {
                    AES cryptPattern = new AES(encryptTextBox.Text);
                    qanak = cryptPattern.DeCrypt(qanak.ToArray()).OfType<byte>().ToList();
                }
                String str = "";
                for (int i = 0; i < 5; i++)
                    str += HelpTools.AutoAddByte(Convert.ToString(qanak[i], 2), 8);
                fileQanakByte = Convert.ToUInt64(str, 2);
                str = "";
                findInfo.RemoveRange((int)fileQanakByte, (int)((UInt64)findInfo.Count - (UInt64)fileQanakByte));
                for (int i = 5; i < 10; i++)
                    str += HelpTools.AutoAddByte(Convert.ToString(qanak[i], 2), 8);
                fileCount = Convert.ToUInt64(str, 2);
                if (checkBox.Checked)
                {
                    findInfo = new AES(encryptTextBox.Text).DeCrypt(findInfo.ToArray()).OfType<byte>().ToList();
                }
                findInfo.RemoveRange(0, 10);
                for (int i = 0; i < (int)fileCount; i++)
                {
                    int nom = 0;
                    int nameLen = findInfo[nom++];

                    str = "";
                    String fileName = "";
                    for (int j = 0; j < nameLen; j++)
                        fileName += Encoding.Unicode.GetString(new byte[2]{
                    Convert.ToByte(HelpTools.AutoAddByte(Convert.ToString(findInfo[nom++], 2), 8),2),
                    Convert.ToByte(HelpTools.AutoAddByte(Convert.ToString(findInfo[nom++], 2), 8),2) });
                    String fileExt = "";
                    int len = findInfo[nom++];
                    for (int j = 0; j < len; j++)
                        fileExt += Encoding.ASCII.GetString(new byte[1] { findInfo[nom++] });
```

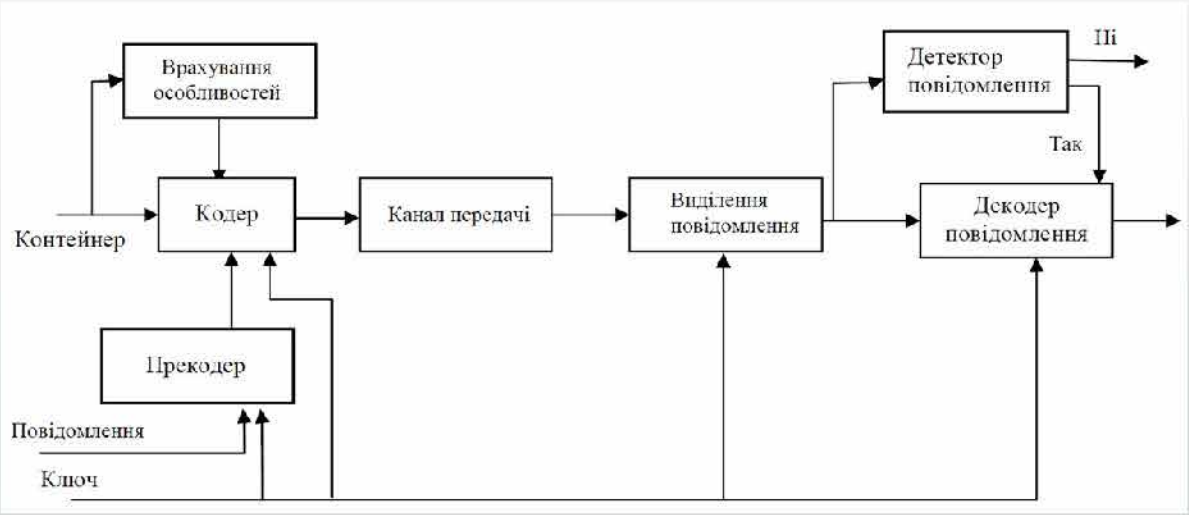
```
                fileExt = "." + fileExt;
                String componentCount = "";
                for (int j = 0; j < 5; j++)
                    componentCount += HelpTools.AutoAddByte(Convert.ToString(findInfo[nom++], 2), 8);
                int compCount = Convert.ToInt32(componentCount, 2);
                byte[] btArray = findInfo.GetRange(nom, compCount).ToArray();
                findInfo.RemoveRange(0, nom + compCount);
                File.WriteAllBytes(savePath + fileName + fileExt, btArray);
            }
        }
        catch
        {
            try{loadingPeocess.Kill();}
            catch (Exception err) { }
            MessageBox.Show("Прихована інформація не знайдена!");
            return;
        }
        try { loadingPeocess.Kill(); }
        catch (Exception err) { }
        MessageBox.Show("Інформація знайдена!");
    }

}
private void BtnContainerRmove_Click(object sender, EventArgs e)
{
    if (containerGridView.SelectedRows.Count < 1)
        return;

    foreach (DataGridViewRow r in containerGridView.SelectedRows)
        if (!r.IsNewRow)
            containerGridView.Rows.RemoveAt(r.Index);
    labelContainer.Text = "Можна приховати: " + GetPoints(GetSize(containerGridView)) + " байтів";
}
private void BtnDataRemove_Click(object sender, EventArgs e)
{
    if (dataGridView.SelectedRows.Count < 1)
        return;
    foreach (DataGridViewRow r in dataGridView.SelectedRows)
        if (!r.IsNewRow)
            dataGridView.Rows.RemoveAt(r.Index);
    labelHide.Text = "Розмір файлу: " + GetPoints(GetSize(dataGridView)) + " байтів";
}
private void CheckBox_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox.Checked)
        encryptTextBox.Enabled = true;
    else
        encryptTextBox.Enabled = false;
}
private void StegPanel_FormClosed(object sender, FormClosedEventArgs e)
{
    foreach (Form form in Application.OpenForms)
    {
        form.Activate();
        form.Show();
        return;
    }
}
private void labelHide_Click(object sender, EventArgs e)
{

}
}
}
```

ДОДАТОК Б

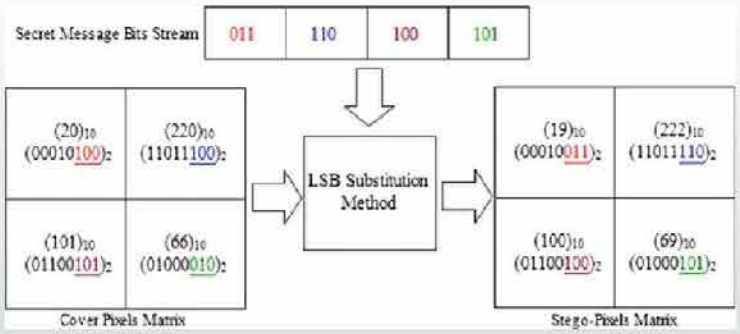
Слайди мультимедійної презентації

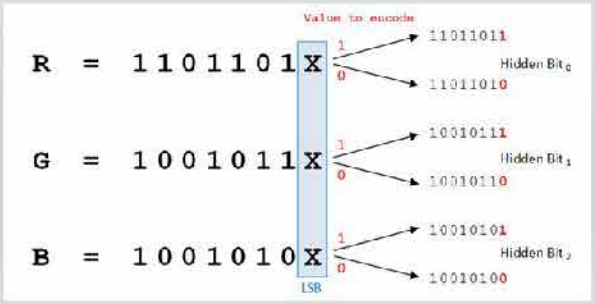
Стеганографічний алгоритм приховування повідомлення

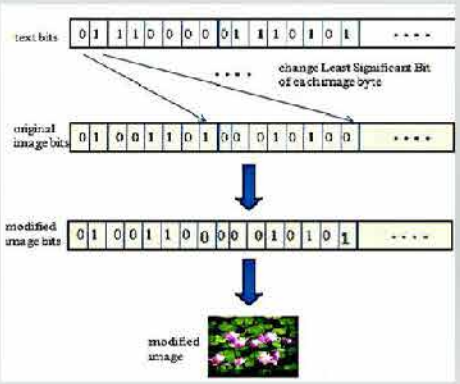
Структурна схема типової стегосистеми



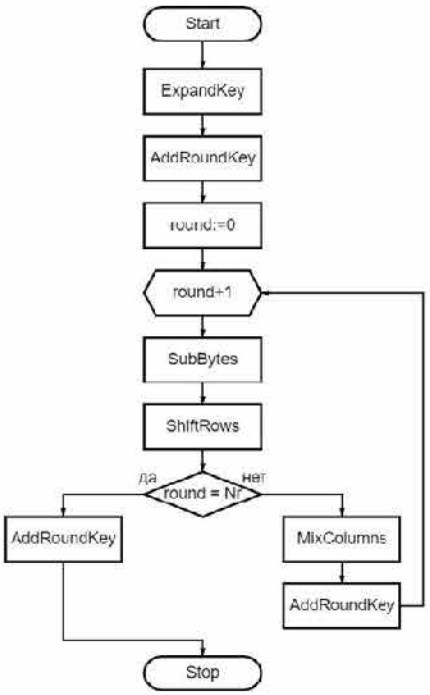
Принцип приховування інформації у зображенні за методом LSB

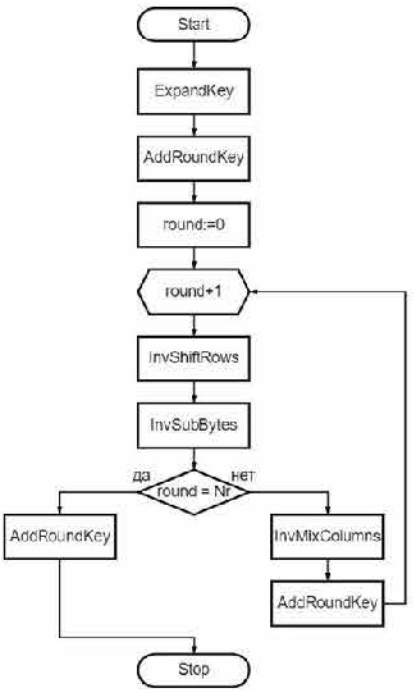




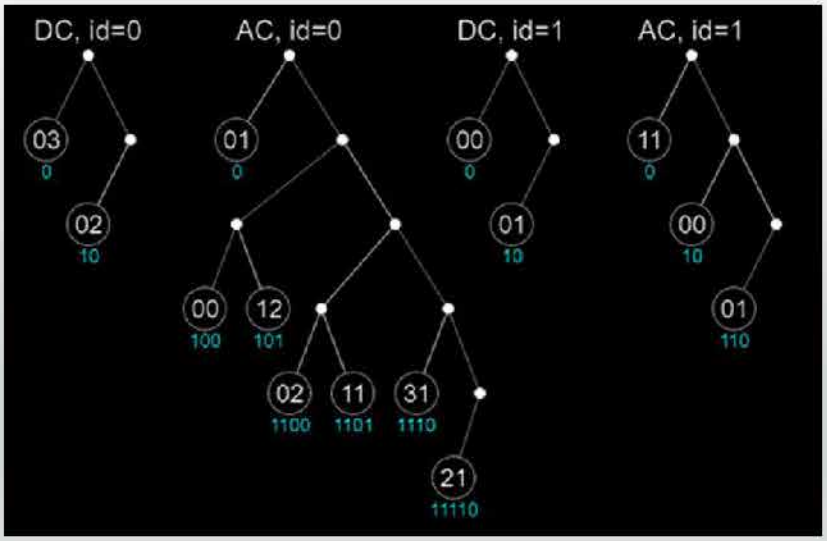


Реалізація шифрування та дешифрування за алгоритмом AES

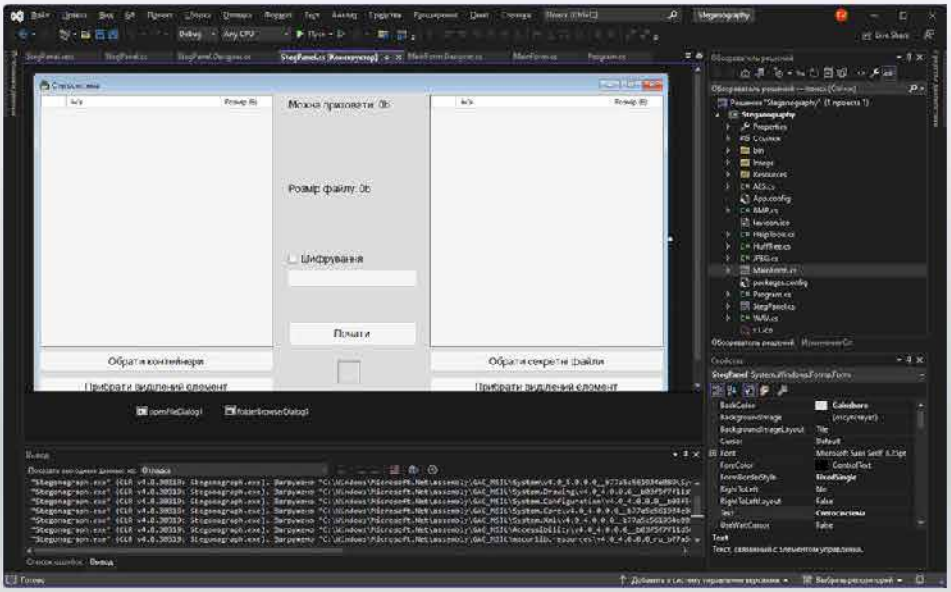




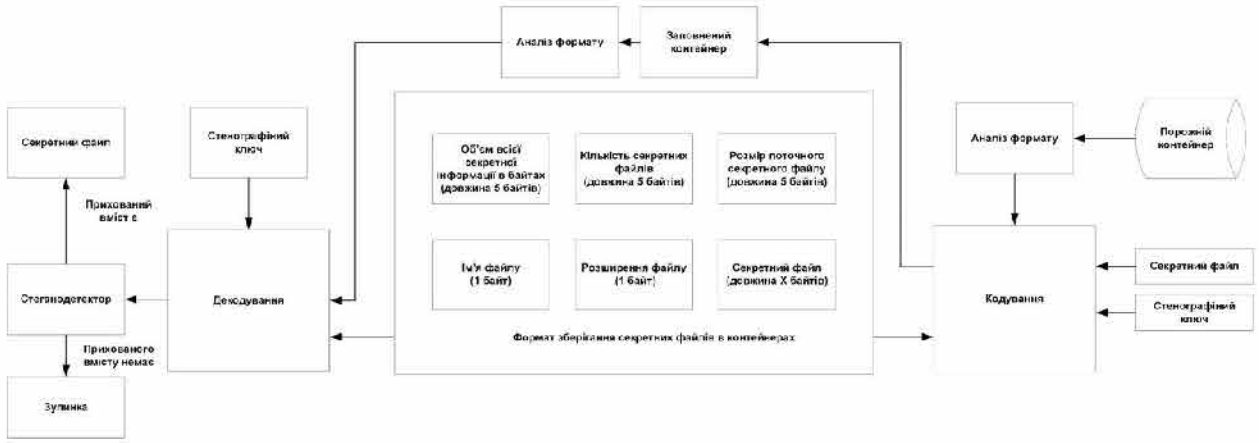
Застосування дерева Хаффмана



Середовище розробки Microsoft Visual Studio 2022



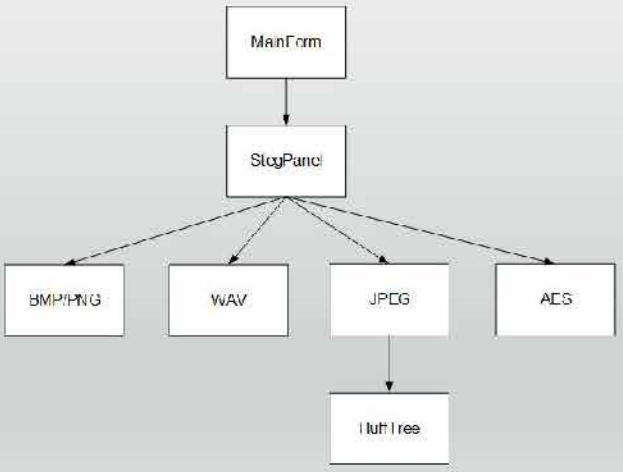
Загальна структура стегосистеми для передачі і захисту прихованих даних



Структура формату зберігання у контейнерах секретних файлів

Формат зберігання секретних файлів в контейнерах					
Об'єм всієї секретної інформації в байтах (довжина 5 байтів)	Кількість секретних файлів (довжина 5 байтів)	Розмір поточного секретного файлу (довжина 5 байтів)	Ім'я файлу (1 байт)	Розширення файлу (1 байт)	Секретний файл (довжина X байтів)
1	2	3	4	5	6

Діаграма взаємодії класів у програмній реалізації стегосистеми



MainForm

Fields:
button1
components
pbHide
pbUnhide
StegonagraphKeyLabel
textBoxStegKey

Methods:
Button1_Click
Dispose
InitializeComponent
KeyStringToLSBByte
MainForm
OpenNewForm
PbHide_Click
PbUnHide_Click
textBoxStegKey_TextChanged
ValidateKey

Складові класу MainForm у застосунку

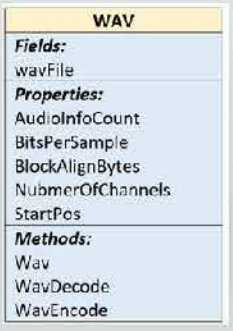
StegPanel

Fields:
btnContainerRmove
bntRemove
btnStart
checkbox
cName
components
ContainerBtn
containerGridView
contCords
cPath
cSize
dataGridView
encryptTextBox
folderBrowserDialog1
labelContainer
labelHide
loadingPeocess
Name
openFileDialoge1
path
pbPicture
SelectItemBtn
Size
StegKey
tpHide

Methods:
BtnContainerRmove_Click
BtnDataRemove_Click
btnStart_Click
CheckBox_ChakedChanged
Dispose
Getpoints
GetSize
InitializeComponent
SelectContainerBtn_Click
SelectDataBtn_Click
StegPanel
StagPamel_FromClosed

Складові класу StegPanel у застосунку

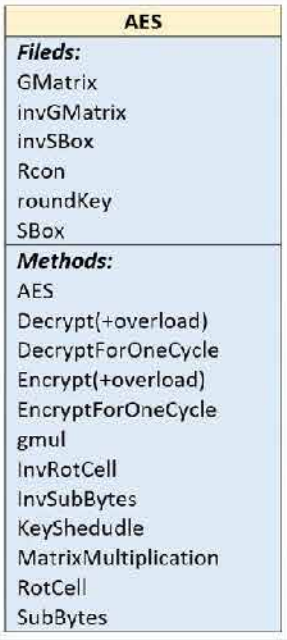
Складові класу BMP/PNG
у застосунку



Складові класу WAV у застосунку

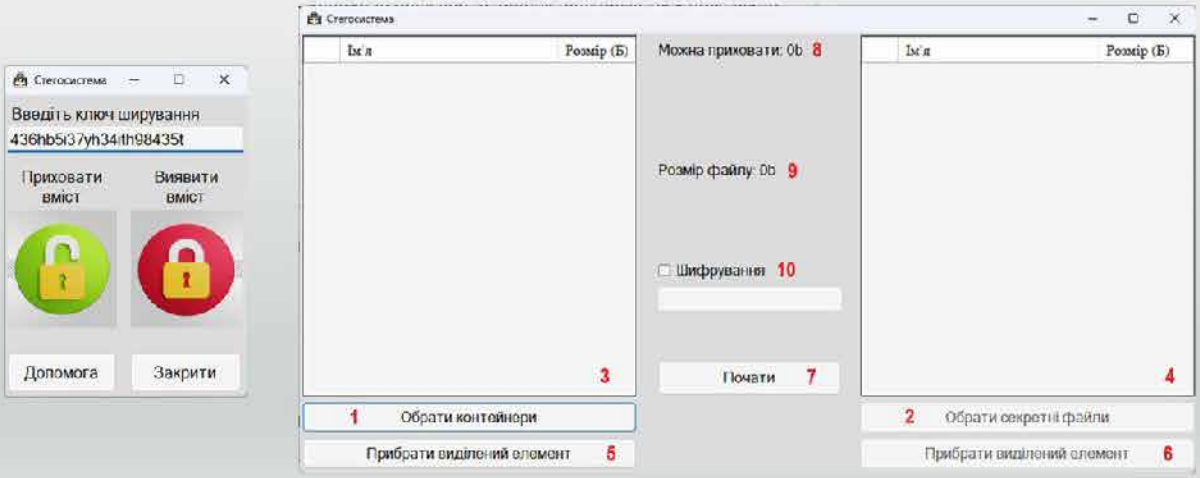
Складові класу JPEG
у застосунку

Складові класу
HuffTree у застосунку

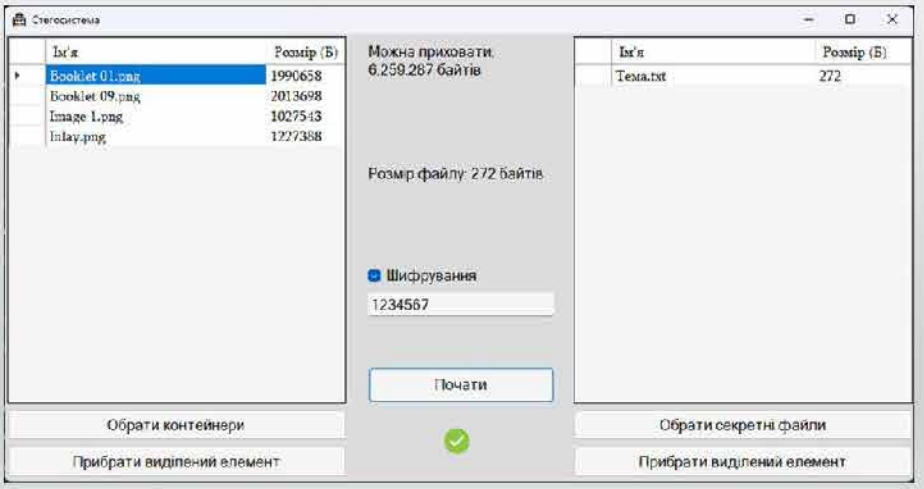


Складові класу AES
у застосунку

Головне та дочірнє вікно програмного застосунку стегосистеми приховування та відновлення інформації



Процес приховування інформації у PNG-контейнері



ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Кекул Дмитра Кириловича

(прізвище, ім`я та по батькові)

Спеціальність: *123 «Комп'ютерна інженерія»*

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

Тема дипломного проекту: *Програмна реалізація стегосистеми для передачі і захисту прихованих даних*

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 85 сторінки. У пояснювальній записці описана програмна реалізація програмна реалізація стегосистеми для передачі і захисту прихованих даних. 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 Кекул Дмитро 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 Кекул Дмитро під час роботи над дипломним проектом вивчив достатньо багато літературних та інтернет-джерел за даною тематикою.*

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

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

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові керівника дипломного проекту *Скорняков В'ячеслав Сергійович*

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

Підпис

«10» 06 2024 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Кекул Дмитра Кириловича

(прізвище, ім'я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

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

Керівник дипломного проекту (роботи) *Скорняков В'ячеслав Сергійович*

(прізвище, ім'я та по батькові)

Тема дипломного проекту (роботи) *Програмна реалізація стегосистеми для передачі і захисту прихованих даних*

Обсяг розрахунково-пояснювальної записки *85* сторінок

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

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

Представлений на рецензію дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений проблемі приховування користувацьких файлів у файлах-контейнерах та складається з пояснювальної записки, додатку з програмним кодом та мультимедійної презентації, що містить приклади роботи програми.

б) характеристика виконання кожного розділу дипломного проекту

Пояснювальна записка складається з основного розділу (аналізу предметної області, проектування застосунку, реалізації застосунку, тестування застосунку), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано. Розділ охорони праці містить загальну інформацію та вимоги до техніки безпеки оператора КТ. Економічний розділ проекту містить розрахунок витрат на НДР та реалізацію проекту.

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 ілюстративні схеми, скріншоти роботи програмного застосунку, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання графічної частини проекту та пояснювальної записки відмінна, розробку виконано у повному обсязі.

г) перелік позитивних якостей дипломного проекту *Обрані стеганографічні алгоритми, зокрема LSB, забезпечують надійне приховування інформації та мінімальне спотворення контейнеру. Програмний засіб дозволяє приховувати інформацію у мультимедійних контейнерах різних видів.*

д) основні недоліки дипломного проекту *1. Бажано було б обмежити вибір файлів-контейнерів у програмному застосунку лише мультимедійними типами даних;*

2. Деякі поля вводу у інтерфейсі програмного застосунку варто було назвати більш відповідно до їх призначення, зокрема шифрування/введіть ключ шифрування, які призначені для різних цілей

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові рецензента *к.т.н. Селіванова Алла Віталіївна*

Місце роботи і посада рецензента *Одеський національний технологічний університет, декан факультету комп'ютерної інженерії, програмування та кіберзахисту*

МІНІСТЕРСТВО ОСВІТИ І НАУКИ ... ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ПРОГРАМУВАННЯ ТА КІБЕРЗАХИСТУ

Підпис: _____

« 17 » червня 2024 р.

UNICHECK
by Turnitin

Ім'я користувача:
Катерина Григоріївна Краснокутська

ID перевірки:
1016234991

Дата перевірки:
07.05.2024 18:53:58 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
07.05.2024 19:54:03 EEST

ID користувача:
100011688

Назва документа: 4КБ-01_Дмитро_Кекул

Кількість сторінок: 64 Кількість слів: 12098 Кількість символів: 86364 Розмір файлу: 4.13 MB ID файлу: 1016015339

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

18.6%
Схожість

Найбільша схожість: 4.49% з Інтернет-джерелом (https://card-file.ontu.edu.ua/server/api/core/bitstreams/6c95086b-bff...

18.6% Джерела з Інтернету | 573 ..... Сторінка 66

Не знайдено джерел з Бібліотеки

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

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

Замінені символи | 54

Підозріле форматування | 13 сторінок

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

Ми, що нижче підписалися,

Кекул Дмитро Кирилович,
здобувач освіти гр. 4КБ-01, та

Скорняков В'ячеслав Сергійович,
керівник дипломного проекту,

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

«Програмна реалізація стегосистеми для передачі і захисту прихованих даних» (автор роботи – Кекул Д.К., керівник роботи – Скорняков В.С.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2024 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 _____ / Кекул Д.К. /

Керівник _____ / Скорняков В.С. /

«10» червня 2024 р.