Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2023 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії”

Виконав здобувач вищої освіти
IV курсу, групи КБ-20-3СК
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Сорокін С.С.
«____» _____ 2023 р.

Керівник проекту
доктор технічних наук, доцент
_____ Коваленко О.В.
«____» _____ 2023 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*

Кафедра *Кібербезпеки та програмного забезпечення*

Освітній ступінь *бакалавр*

Галузь знань ._12 _____*"*Інформаційні технології*"*_____

Спеціальність *125 "*Кібербезпека*"*

Освітньо-професійна (освітньо-наукова) програма _*"*Кібербезпека*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 17 » січня 2023 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ПЕРШИМ (БАКАЛАВРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Сорокіну Сергію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії*

2. Керівник роботи *Коваленко Олександр Володимирович, докт. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 13-02 від 5.01.2023 року

3. Строк подання студентом роботи до захисту *23.05.2023 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи кібербезпеки в промислову експлуатацію.
6. Висновки

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Структурна схема системи кібербезпеки	*1 аркуш*
Функціональна схема системи кібербезпеки	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

7. Дата видачі завдання « 17 » січня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.03.2023 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2023 р.	
3.	Розробка моделі компонента	20.03.2023 р.	
4.	Розробка структур даних	25.03.2023 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2023 р.	
6.	Програмування алгоритмів	10.04.2023 р.	
7.	Оформлення ПЗ	17.04.2023 р.	
8.	Попередній захист роботи	23.05.2023 р.	

Дата видачі завдання
« 17 » січня 2023 р.

Підпис керівника
_____ Коваленко О.В.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2023 р.

Підпис здобувача
_____ Сорокін С.С.
(прізвище та ініціали)

АНОТАЦІЯ

Сорокін С.С. Програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії. 125 Кібербезпека. Центральноукраїнський національний технічний університет. Кропивницький. 2023.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

Метою розробки є програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

Результат роботи – програмна реалізація системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10/11.

Програму розроблено в середовищі Delphi 10.4.1.

Ключові слова: кібербезпека, стеганографія

ABSTRACT

Sorokin S.S. Cybersecurity system software for protecting confidential information on the network using steganography. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2023.

In this graduation thesis for the first (bachelor) level of higher education, software is developed, which is intended for a cyber security system to protect confidential information on the network using steganography.

The purpose of the development is the software of the cyber security system for the protection of confidential information in the network by the method of steganography.

The result of the work is the software implementation of the cyber security system for the protection of confidential information in the network by the method of steganography.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on PCs of IBM PC architecture with Windows 10/11 OS.

The program was developed in the Delphi 10.4.1 environment.

Keywords: cyber security, steganography

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ВДТ	–	відео-дисплейні термінали
ЕОМ	–	електронно-обчислювальна машина
ЕПТ	–	електроннопроменева трубка
ЕЦП	–	електронний цифровий підпис
ЗІ	–	захист інформації
ПЗ	–	програмне забезпечення
ПК	–	персональний комп’ютер
СБ	–	служба безпеки
ТЗ	–	технічне завдання
ЦВЗ	–	цифрові водяні знаки
DES	–	стандарт шифрування США
DSA	–	Digital Signature Algorithm
ECDSA	–	Elliptic Curve Digital Signature Algorithm
EGSA	–	El Gamal Signature Algorithm
IDEA	–	International Date Encryption Algorithm – алгоритм шифрування
IP	–	Internet Protocol
LSB	–	Least Significant Bits – метод стеганографії
PGP	–	Pretty Good Privacy – міжнародний криптографічний стандарт
RSA	–	алгоритм асиметричного шифрування
SHA-1	–	Secure Hash Algorithm 1 – алгоритм криптографічного хешування
TDES	–	Triple DES –модифікація DES з трьома незалежними підключами
JPEG	–	Joint Photographic Experts Group – растровий формат зображення

ВСТУП

Актуальність теми. Стеганографія – це техніка приховування секретних даних у звичайному, несекретному файлі чи повідомленні, щоб уникнути виявлення; потім секретні дані витягуються в місці призначення. Використання стеганографії можна поєднати з шифруванням як додатковий крок для приховування або захисту даних.

Стеганографію можна використовувати для приховування майже будь-якого типу цифрового вмісту, включаючи текст, зображення, відео чи аудіоконтент; дані, які потрібно приховати, можуть бути приховані майже в будь-якому іншому типі цифрового вмісту. Вміст, який потрібно приховати за допомогою стеганографії – так званий *прихований текст* – часто шифрується перед включенням у нешкідливий *текстовий* файл обкладинки або потік даних. Якщо прихований текст не зашифрований, він зазвичай певним чином обробляється, щоб ускладнити виявлення секретного вмісту.

Стеганографією займаються ті, хто хоче передати секретне повідомлення або код. Хоча існує багато законних способів використання стеганографії, розробники зловмисного програмного забезпечення також використовують стеганографію для приховування передачі шкідливого коду.

Форми стеганографії використовувалися протягом століть і включають майже будь-які методи приховування секретного повідомлення в нешкідливому контейнері. Наприклад, використання невидимого чорнила для приховування секретних повідомлень у необразливих повідомленнях; приховування документів, записаних у вигляді мікроточок, діаметр яких може досягати 1 міліметра, на або всередині начебто легітимної кореспонденції; і навіть за допомогою багатокористувацьких ігрових середовищ для обміну інформацією.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем для захисту конфіденційної інформації у мережі методом стеганографії.

– Дослідження системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

– Програмна реалізація системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі для захисту конфіденційної інформації у мережі методом стеганографії.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

У сучасній цифровій стеганографії дані спочатку шифруються або обфусцуються іншим способом, а потім вставляються за допомогою спеціального алгоритму в дані, які є частиною певного формату файлу, наприклад зображення JPEG , аудіо- чи відеофайл. Секретне повідомлення можна вставити у звичайні файли даних різними способами. Одним з методів є приховування даних у бітах, які представляють пікселі одного кольору, що повторюються в рядку у файлі зображення. Застосовуючи зашифровані дані до цих надлишкових даних у якийсь непомітний спосіб, результатом буде файл зображення, який виглядає ідентичним оригінальному зображенню, але містить шаблони «шуму» звичайних незашифрованих даних.

Практика додавання водяного знака – торгової марки або інших ідентифікаційних даних, прихованих у мультимедійних або інших файлах вмісту – є одним із поширених способів використання стеганографії. Водяні знаки – це техніка, яка часто використовується онлайн-видавцями для ідентифікації джерела медіафайлів, які, як було виявлено, поширюються без дозволу.

Хоча існує багато різних способів використання стеганографії, включаючи вбудовування конфіденційної інформації в типи файлів, одним із найпоширеніших методів є вбудовування текстового файлу у файл зображення. Після цього будь-хто, хто переглядає файл зображення, не зможе побачити різницю між оригінальним файлом зображення та зашифрованим файлом; це досягається шляхом збереження повідомлення з менш значущими бітами у файлі даних. Цей процес можна завершити вручну або за допомогою інструменту стеганографії.

Які переваги стеганографії над криптографією?

Стеганографія відрізняється від криптографії, але використання обох разом може допомогти покращити безпеку захищеної інформації та запобігти виявленню таємного зв'язку. Якщо стеганографічно приховані дані також зашифровані, дані все ще можуть бути захищені від виявлення, хоча канал більше не буде захищений від виявлення. Використання стеганографії в поєднанні з шифруванням має переваги перед зв'язком лише за допомогою шифрування.

Основна перевага використання стеганографії для приховування даних перед шифруванням полягає в тому, що вона допомагає приховати факт прихованих конфіденційних даних у файлі чи іншому вмісті, який містить прихований текст. У той час як зашифрований файл, повідомлення або корисне навантаження мережевого пакета чітко позначені та ідентифіковані як такі, використання стеганографічних методів допомагає приховати присутність безпечного каналу.

Програмне забезпечення для стеганографії

Програмне забезпечення стеганографії використовується для виконання різноманітних функцій, щоб приховати дані, включаючи кодування даних, щоб підготувати їх до приховування в іншому файлі, відстеження того, які біти текстового файлу обкладинки містять приховані дані, шифрування даних для бути прихованим і видобувати приховані дані його призначеним одержувачем.

Існують пропрієтарні програми, а також програми з відкритим кодом та інші безкоштовні програми для створення стеганографії. OpenStego – програма стеганографії з відкритим кодом; інші програми можна охарактеризувати типами даних, які можна приховати, а також типами файлів, у яких ці дані можуть бути приховані. Деякі онлайн-інструменти стеганографії включають Xiao Steganography, який використовується для приховування секретних файлів у зображеннях BMP або WAV; Стеганографія зображень, інструмент Javascript, який приховує зображення в інших файлах зображень; і Crypture, інструмент командного рядка, який використовується для виконання стеганографії.

1.2 Область застосування

Залежно від характеру об'єкта прикриття (фактичного об'єкта, в який вбудовані секретні дані), стеганографія може бути розділена на п'ять типів:

1. Стеганографія тексту.
2. Стеганографія зображення.
3. Стеганографія відео.
4. Аудіо стеганографія.
5. Мережева стеганографія.

Розглянемо кожен з них докладніше.

Стеганографія тексту

Текстова стеганографія приховує інформацію всередині текстових файлів. Це включає в себе такі речі, як зміна формату існуючого тексту, зміна слів у тексті, генерація випадкових послідовностей символів або використання контекстно-вільної граматики для створення читабельних текстів. Для приховування даних у тексті використовуються такі різні методи:

– Метод на основі формату.
– Випадкова та статистична генерація.
– Лінгвістичний метод.

Стеганографія зображення

Приховування даних шляхом використання обкладинки як зображення називається стеганографія зображення. У цифровій стеганографії зображення є широко використовуваним джерелом покриття, оскільки в цифровому представленні зображення присутній величезна кількість бітів. Існує багато способів приховати інформацію всередині зображення. Загальні підходи включають:

– Вставка найменшого біта.
– Маскування та фільтрація.
– Надлишкове кодування шаблону.

– Шифрувати та розсіювати.

– Кодування та косинусне перетворення.

Аудіо стеганографія

В аудіостеганографії секретне повідомлення вбудовано в аудіосигнал, який змінює двійкову послідовність відповідного аудіофайлу. Приховування секретного повідомленняges in дігітал такі є набагато більшою різницеюскладний процес у порівнянні з іншими, такими як стеганографія зображення. Різні методи аудіостеганографії включають:

– Кодування найменш значущих бітів.

– Парність кодування.

– Фазове кодування.

– Розширений спектр.

Цей метод приховує дані у звукових файлах WAV, AU і навіть MP3.

Стеганографія відео

У Video Steganography ви можете приховати тип даних у формат цифрового відео. Перевагою цього типу є велика кількість даних, які можна приховати всередині, і те, що це рухомий потік зображень і звуків. Ви можете розглядати це як поєднання стеганографії зображень і стеганографії аудіо. Два основні класи відеостеганографії включають:

– Вбудовування даних у нестиснене необроблене відео та їхнє стиснення пізніше.

– Вбудовування даних безпосередньо в стиснений потік даних.

Стеганографія мережі (стеганографія протоколу)

Це техніка вбудовування інформації в мережевий контроль протоколи, що використовуються для передачі даних, такі як TCP, UDP, ICMP тощо. Ви можете використовувати стеганографію в деяких прихованих каналах, які можна знайти в моделі OSI. Наприклад, ви можете приховати інформацію в заголовку TCP/IP.

У сучасному цифровому світі для стеганографії доступні різноманітні програмні засоби. У решті цього підручника зі стеганографії ми розглянемо деякі популярні стеганографічні інструменти та їхні можливості.

Переваги стеганографії

Стеганографія – це метод, який дозволяє легко приховати повідомлення в іншому, щоб зберегти його в секреті. У результаті приховане повідомлення залишається прихованим. Стеганографічний підхід може принести користь зображенням, відео та аудіофайлам. Додаткові переваги включають:

– На відміну від інших методів, стеганографія має додаткову перевагу приховування комунікацій настільки добре, що вони не привертають уваги. Однак у країнах, де шифрування є незаконним, надсилання зашифрованого повідомлення, яке можна легко розшифрувати, викликає підозри та може бути ризикованим.

– Стеганографія – це форма шифрування, яка захищає інформацію в повідомленні та з'єднання між відправником і одержувачем.

– Три важливі елементи стеганографії – безпека, місткість і надійність – роблять доцільним приховану передачу інформації за допомогою текстових файлів і розробку прихованих каналів зв'язку.

– Ви можете зберігати зашифровану копію файлу, що містить конфіденційну інформацію, на сервері, не побоюючись, що неавторизовані особи отримають доступ до даних.

– Урядові та правоохоронні органи можуть таємно спілкуватися за допомогою стеганографії корпорацій.

– Отримайте навички, щоб пройти співбесіду з кібербезпеки

Використання стеганографії для здійснення атак

Зараз атаки зазвичай автоматизуються за допомогою сценаріїв PowerShell або BASH. І хакери теж. Документи Excel і Word із увімкненими макросами були поширеним вектором атак. Прихований сценарій запускається, коли ціль відкриває шкідливий файл Word або Excel.

Зловмисник може отримати доступ до системи, не змусивши жертву встановити Steghide. Зловмисник використовує стеганографічну програму, щоб скористатися такими поширеними інструментами Windows, як Excel і PowerShell. Як тільки жертва прочитає документ, хакеру стане легше атакувати систему.

Штучний інтелект і стеганографія

Хакери також використовують штучний інтелект (ШІ). Стеганографія є лише одним із багатьох методів, які штучний інтелект все частіше використовує для приховування своєї діяльності. Реалізації штучного інтелекту налаштували навіть стеганографічні методи, щоб ускладнити виявлення атак.

Виявлення стеганографії

У своїй роботі аналітики безпеки шукають індикатори стандартних стратегій тестування атак і проникнення (TTP). Загальні підписи, які використовуються стеганографічними програмами, були виявлені з часом. Через це, наприклад, антивірусне програмне забезпечення може легко помітити типову поведінку стеганографічних програм.

У результаті тестери на проникнення та зловмисники постійно коригують свої методи, щоб залишатися непоміченими. Подібним чином дослідники безпеки постійно шукають нові сигнатури та тактики атак, а кіберзлочинці постійно адаптують свої інструменти та підходи.

Реальні атаки з використанням стеганографії

У 2020 році компанії у Великій Британії, Німеччині, Італії та Японії постраждали від кампанії з використанням стеганографічних документів.

Хакери можуть уникнути виявлення, використовуючи стеганографічне зображення, завантажене на хорошу платформу, як-от Imgur, для зараження документа Excel. Mimikatz, зловмисне програмне забезпечення, яке викрадає паролі Windows, було завантажено за допомогою секретного сценарію, включеного до зображення.

Пом'якшення атак на основі стеганографії

Стеганографію легко реалізувати під час кібератаки. Однак цьому набагато важче запобігти, оскільки люди, які становлять загрозу, стають більш винахідливими та винахідливими, що ускладнює розробку контрзаходів.

Код, замаскований у зображеннях та інших видах обфускації, швидше за все, буде динамічно виявлений механізмом поведінки. Тому компаніям слід використовувати сучасні рішення для захисту кінцевих точок, які виходять за рамки статичних перевірок, елементарних підписів та інших старомодних компонентів.

Співробітники повинні знати про ризик відкриття файлів зображень, оскільки вони можуть містити віруси. Крім того, найновіші патчі безпеки слід встановлювати щоразу, коли вони стають доступними, а фірми повинні використовувати веб-фільтрацію, щоб гарантувати, що їхні співробітники можуть безпечно переглядати веб-сторінки.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти

Існує багато доступного програмного забезпечення, яке пропонує стеганографію. Деякі пропонують звичайну стеганографію, але деякі пропонують шифрування перед приховуванням даних. Це інструменти стеганографії, які доступні безкоштовно:

– Stegosuite – це безкоштовний інструмент стеганографії, написаний мовою Java. За допомогою Stegosuite ви можете легко приховати конфіденційну інформацію у файлах зображень.

– Steghide – це програмне забезпечення стеганографії з відкритим кодом, яке дозволяє приховати секретний файл у зображенні чи аудіофайлі.

– Xiao Steganography – це безкоштовне програмне забезпечення, яке можна використовувати для приховання даних у зображеннях BMP або у файлах WAV.

– SSuite Picsel – ще одна безкоштовна портативна програма для приховання тексту у файлі зображення, але вона має інший підхід порівняно з іншими інструментами .

– OpenPuff – це професійний стеганографічний інструмент, у якому можна зберігати файли зображень, аудіо, відео чи флеш-файли

– Стеганографія зображень: ця програма є інструментом JavaScript, який використовується для приховування зображень в інших файлах зображень

– OpenStego: ця програма є інструментом стеганографії з відкритим кодом

– Стеганографія Xiao: Xiao приховує секретні файли у файлах WAV або BMP

– Crypture: Ця програма є інструментом командного рядка, який використовується для проведення стеганографії

– NoClue: ця програма є інструментом із відкритим вихідним кодом, який приховує текстову інформацію у файлах-носіїх відео та зображень

– Steganography Master: ця програма – це інструмент із відкритим вихідним кодом на базі Android, який може приховувати текст у зображенні та надає вам інструмент декодування для отримання прихованих текстових повідомлень із файлів зображень. Він підтримує кілька форматів зображень (BMP, JPG, ICO, PNG)

– Steghide: Steghide – це програма, яка приховує дані в різних аудіофайлах і файлах зображень, включаючи JPEG, BMP, AU і WAV

Що ж, це кілька інструментів для виконання стеганографії. Є багато інших різних інструментів з різними можливостями. Однак ви отримаєте бажані результати від цих засобів.

Які характеристики програмного забезпечення стеганографії

У програмному забезпеченні стеганографії його можна поєднати з поточними методами зв'язку, стеганографію можна використовувати для видачі прихованих обмінів.

Основною метою стеганографії є безпечне спілкування у абсолютно невизначений спосіб і запобігання виклику підозр щодо передачі прихованої інформації.

Існує кілька програм, які мають численні вимоги до використовуваних методів стеганографії. Наприклад, деяким програмам може знадобитися безумовна невидимість приватної інформації, у той час як інші потребують більш високого секретного повідомлення, щоб бути прихованим.

Це не для того, щоб утримати інших від розуміння прихованої інформації, але це може утримати інших від думки, що інформація взагалі існує. Якщо стеганографічний підхід спонукає когось запідозрити носій, це означає, що підхід відхиляється.

Стеганографічне програмне забезпечення дозволяє приховувати дані в графічних, звукових і, здавалося б, порожніх носіях. Наприклад, він містить дані, надіслані через зображення та малюнки.

Потрійні RGB використовують 3 байти на піксель. Коли він може вставити дані в 24-розрядне зображення, контейнерний документ виглядає необразливо, і людське око не може проаналізувати вбудований файл повідомлення. Це може захистити нас від проблем стиснення файлів, коли зберігання даних може бути перешкоджано.

Стеганографією зображень найефективніше керувати програмним забезпеченням JPEG. Це може означати, що вихідний код підтримується для компіляції коду на кількох платформах.

JPEG використовує кодування з втратами для стиснення інформації. Для виведення використовуються файли JFIF. JFIF включає етапи як із втратами, так і без втрат. Інформація, яку потрібно передати, прихована серед цих фаз.

Стиснення файлів у форматі JPEG є його найбільшою перевагою. Великі зображення в невизначених кольорах можна зберігати у відносно невеликих файлах. Інший екземпляр може містити дані, надіслані через звукові чи аудіофайли.

У галузі є кілька пакетів стеганографічного програмного забезпечення, доступних зовсім недавно, і містять Hide-&-Seek, StegosDos, White Noise тощо. У програмному забезпеченні стеганографії повідомлення шифруються перед вставленням, а також для забезпечення більшого рівня захисту.

Основною характеристикою стеганографії є той факт, що дані приховані в надлишкових бітах об’єкта, а оскільки зайві біти пропускаються під час використання JPEG, існує побоювання, що приховане повідомлення може бути втрачено.

У JPEG його можна використовувати для створення невидимих для людського ока змін зображення. Під час процедури перетворення DCT алгоритму

стиснення в інформації про коефіцієнти з’являються помилки округлення, які не є очевидними.

Хоча ця функція є тим, що вона може визначити алгоритм як втрачений, цю функцію також можна використовувати для приховування повідомлень.

Cipher Image Free

Безкоштовна програма Cipher Image Free дозволяє зберігати відразу кілька зашифрованих зображень (стеганографія) в одному багатосторінковому TIFF файлі. Вона підтримує більше 21 форматів графічних документів, має простий і інтуїтивно зрозумілий інтерфейс.

Рисунок 2.1 – Інтерфейс користувача Cipher Image Free

Особливості програми:

– 7 форматів для збереження файлів.

– Схований текст повідомлення може бути до 64 Кбайт.

– 128-бітне шифрування зображень із випадковим ключовим словом.

– Убудований генератор ключових слів Підказки при роботі «Magic Help».

– Підтримка всіх версій Windows.

Ultima Steganography

Ultima Steganography це дуже простий інструмент стеганографії для Windows. За допомогою цього інструмента ви зможете зашифрувати файл і сховати його в зображенні. Після того, як файл захований у зображення, збережене зображення поводиться як звичайна картинка, так само відкривається й завантажується. Єдина відмінність даного файлу від звичайного зображення це те, що в ньому захований файл. Схоронність і цілісність файлів є найважливішими питаннями у світі цифрових технологій.

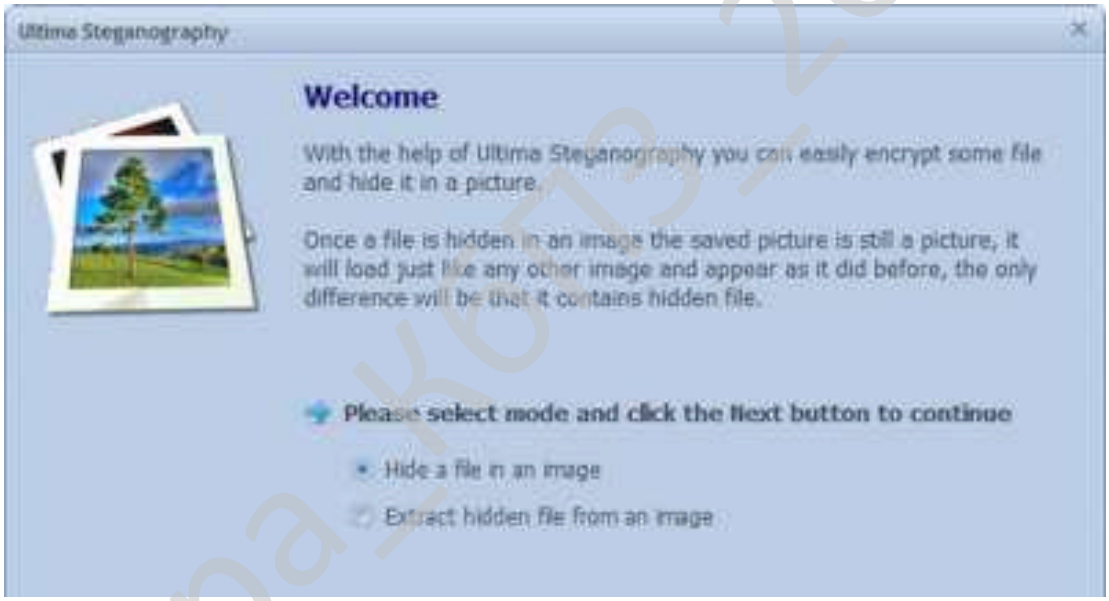

Рисунок 2.2 – Інтерфейс користувача Ultima Steganography

Max File Encryption

Max File Encryption це зручна й потужна програма для шифрування файлів, що забезпечить надійний захист Ваших даних. За допомогою Max File Encryption, Ви зможете зашифрувати файли будь-якого типу (включаючи документи Microsoft Word, Excel і PowerPoint), сховати файли (використовуючи метод стеганографії), а також, створити пакети, що самедешифруються.

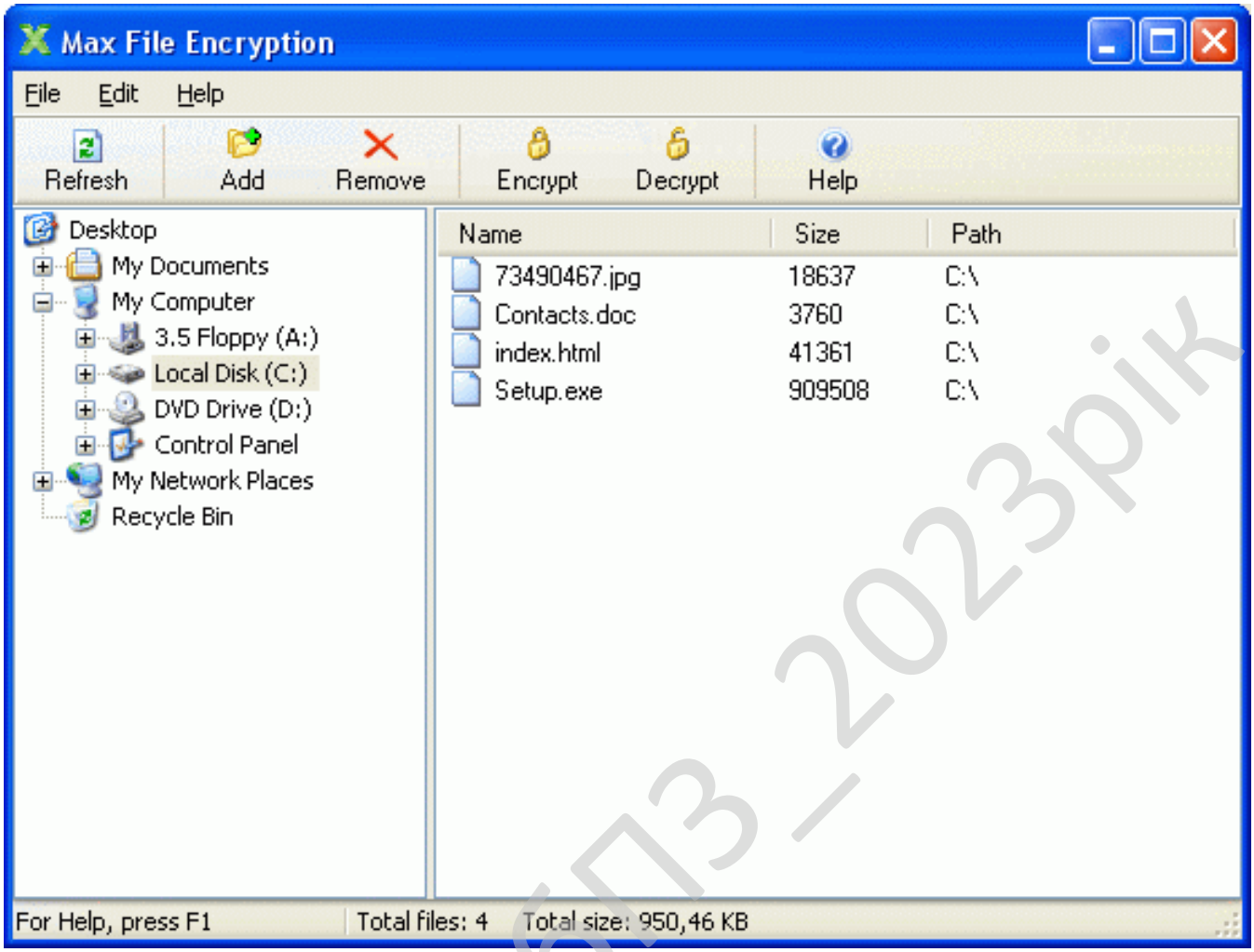

Рисунок 2.3 – Інтерфейс користувача Max File Encryption

Програма використовує потужний і надійний алгоритм шифрування Blowfish, що гарантує безпека Ваших даних. Пакети, що самодешифруються, можуть бути розшифровані й на тих комп'ютерах, де програма Max File Encryption не встановлена. Використовуючи метод стеганографії, Ви зможете сховати дані в самих звичайних файлах (носіях), таких як графічні, аудіо, відео або, навіть, у додатках і dll файлах. При використанні стеганографії файл-носій залишається повністю працездатним! Програма має інтуїтивно зрозумілий і зручний інтерфейс, що дозволяє будь-якому користувачеві, що навіть не має знань в області криптографії, надійно захистити свої дані.

Основні характеристики програми:

– Шифрування файлів – програма робить шифрування файлів за допомогою надійного й популярного алгоритму шифрування Blowfish.

– Пакети, що самодешифруються, – програма дозволяє створювати саме– пакети, щодешифруються, які можна дешифрувати без програми Max File Encryption на будь-якому комп'ютері.

– Стеганографія – Max File Encryption підтримує стеганографію (шифрування й приховання динячих у звичайних файлах).

– Підтримка файлів будь-якого типу – програма дозволяє шифрувати будь-які типи файлів, включаючи документи Microsoft Word, Excel і PowerPoint.

– Підтримка файлів великого розміру (до 4 Гб).

– Можливість модифікувати зашифровані файли.

– Підтримка ОС Windows 10/11.

– Зручний і легкий інтерфейс.

2.2 Обґрунтування вибору засобів для побудови системи кібербезпеки та мови програмування

Embarcadero Delphi, раніше Borland Delphi і Codegear Delphi, – інтегроване середовище розробки ПЗ для Microsoft Windows, Mac OS, iOS і Android мовою Delphi (що раніше носила назву Object Pascal), створена спочатку фірмою Borland і на даний момент приналежна й розроблювальна Embarcadero Technologies. Embarcadero Delphi є частиною пакета Embarcadero RAD Studio і поставляється в чотирьох редакціях: Community (поширюється безкоштовно й має обмежену ліцензію на використання в комерційних цілях), Professional, Enterprise і Architect.

Delphi 10.4 Sydney

Випущено 26 травня 2020 року. RAD Studio Delphi 10.4 забезпечує значно поліпшену високопродуктивну нативну підтримку Windows, кращу продуктивність розробки, миттєві підказки code completion, прискорення

виконання коду із синтаксисом керованих записів, поліпшення виконання паралельних завдань на сучасних багатоядерних CPU, а також містить більш 1000 виправлень багів, поліпшення продуктивності середовища й бібліотек і багато чого крім того.

Основні можливості Delphi 10.4.1:

− Істотні розширення для Windows: поліпшення для застосунків на моніторах 4K High DPI, інтеграція з новим WebView2 на базі Chromium, використання розширених title bars, таких же, як в Office, Explorer, Google Chrome.

− Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

− Істотне поліпшення Delphi Code Insight (без можливого блокування IDE – в окремому процесі), що допоможе при роботі з великими проектами.

− Тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання.

− Розширена підтримка бібліотек C++: ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode.

− Відладник Win 64 (на LLDB) і збирач для C++.

− Поліпшення для C++: Включена велика кількість поліпшень STL з Dinkumware.

− Підтримка Metal Driver GPU для macOS і iOS.

− Вбудований Fmxlinux.

− Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API. Реалізація компонента Media Player для macOS тепер використовує Avfoundation. Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

− Численні поліпшення швидкості й стабільності роботи нашої бібліотеки The Parallel Programming Library (PPL).

− Додані оновлені драйвери для FireBird, PostgreSQL і SQLite.

− Клієнтські бібліотеки HTTP і REST Client розширені застосунковими можливостями роботи з HTTPS. Також були розширені можливості підтримки Amazon AWS services

− У технологію Visual LiveBindings внесена безліч поліпшень, у тому числі швидкодії, що стосуться, застосунків на VCL і FireMonkey

RAD Studio 10.4 Короткий огляд:

– Істотні розширення для Windows. Створення застосунків, що чудово виглядають, із чіткими елементами інтерфейсу на 4k моніторах High DPI за допомогою нової гнучкої підтримки стилів елементів керування на екрані. Інтеграція із сучасними, безпечними web-технологіями від Microsoft – новим WebView2 на базі Chromium. Використання сучасних розширених title bars, таких же, як в Office, Explorer, Google Chrome, у своїх проектах. Істотні поліпшення надійності налагодження в новому відладнику для C++ Windows 64-bit.

– Зросла продуктивність розробки. Ріст продуктивності за рахунок миттєвої реакції підказок code completion у середовищі IDE. Краща сумісність із уже наявною кодовою базою, і спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю. Швидке зв'язування даних і візуальних елементів за допомогою розширеної технології Visual LiveBindings з підвищеною швидкодією. Просте використання розповсюджених бібліотек C++, наприклад, ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode. Оновлена підтримка Amazon AWS cloud.

– Поліпшення швидкодії і якості. Більш 1000 поліпшень швидкодії і якості. Краща ефективність коду за допомогою нового синтаксису custom managed records. Більш швидке виконання паралельних завдань на сучасних багатоядерних CPU. Переконаєтеся в прискоренні відображення на екрані з підтримкою Metal API на macOS і iOS. Краща сумісність із уже наявною кодовою базою й спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю.

Істотне поліпшення Delphi Code Insight

Як найбільше й головне поліпшення інструментів програмування Delphi за багато років, в 10.4 Delphi Code Insight реалізований через Language Server Protocol (LSP). LSP – це технологія генерації результатів для code completion, навігації й інших сервісів в окремому процесі. Це значить, що code completion і Code Insight одержать більш точні результати без блокування IDE. 10.4 забезпечує набагато більш високу продуктивність розроблювачів, які працюють із більшими проектами, що містять мільйони рядків коду.

Delphi Custom Managed Records

Ключове розширення мови Delphi: тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання. Управляйте тем, як ці структури створюються, копіюються й звільняються з допомогу вашого коду, який буде виконуватися у відповідний момент.

Це розширює потужність конструкцій records в Delphi, які використовуться щоб одержати більшу ефективність у порівнянні із класами.

Єдине керування пам'яттю

Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

У порівнянні з Automatic Reference Counting (ARC), це дає кращу сумісність із існучим кодом і спрощує написання компонентів, бібліотек і застосунків.

ARC модель керування пам'яттю model залишилася для керування рядками й посиланнями на тип інтерфейсу на всіх платформах. Для C++ це означає, що при створенні й звільненні Delphi-style класів в C++ використовується звичайне керування пам'яттю, як у будь-якого heap-allocated класу C++, що значно знижує складність коду.

Розширена підтримка бібліотек C++

В 10.4 ми портували багато популярних бібліотек C++ у C++Builder.

Забезпечивши оптимізовану підтримку бібліотек ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode, поряд із уже підтримуваними Boost і Eigen, які можуть бути додані за допомогою менеджера пакетів Getit.

Win 64-відладник і збирач для C++

В 10.4 з'явився новий відладник C++ для Windows 64-bit. Відладник заснований на LLDB і показує значне збільшення стабільності при налагодженні 64-bit застосунків поряд з новими відладочними можливостями, такими як перегляд і інспекція типів начебто рядків C++ і Delphi, а також колекцій STL, включаючи std::vector, std::map і інших. Крім того, згенерована для застосунку відладочна інформація має інший внутрішній формат, сприяючи більш стабільному й багатому на можливості процесу налагодження, більш докладним перегляду й інспекції в debug-time.

Підвищення якості й швидкодії інструментів

- Велика кількість поліпшень STL від Dinkumware.
- Поліпшені деякі найважливіші методи й області RTL, на базі поліпшень сумісності з популярними бібліотеками C++.
- Поліпшена підтримка Cmake.
- Велика кількість виправлень для підвищення стабільності і якості.
- Відновлення Windows API – Обновлено й додали безліч декларацій API щоб добитися ще більшої інтеграції із платформою Windows.
- Загальні вдосконалення в бібліотеці доступу до БД FireDAC, включаючи оновлені драйвера для FireBird, PostgreSQL і SQLite. Вибір статичного або динамічного підключення SQLite до застосунку.

Змінені стилі VCL для High DPI

В 10.4, архітектура стилізації VCL була суттєво розширена для підтримки High DPI і 4K моніторів. Тепер усі елементи UI на формі VCL автоматично масштабуться під відповідне до монітора дозвіл для показу форми. Був оновлений API стилізації для підтримки стилів high DPI.

Кожний графічний елемент UI може бути обраний з наборів різних масштабів і масштабований до потрібного DPI, що дає чітке зображення елементів UI на всіх моніторах.

Нові High DPI стилі й стилізація окремих VCL компонент

Обновлено велике число вбудованих і преміальних VCL стилів для підтримки нового режиму стилізації High-dpi. Це дозволяє вам створювати застосунку з відмінним дизайном для всіх моніторів.

Розроблювачі VCL застосунків тепер можуть використовувати трохи VCL стилів на різних формах в одному застосунку або в різних компонентів на одній формі. Це також включає стилізацію компонентів загальною темою для платформи. Крім застосункової гнучкості використання стилів, це дозволяє використовувати нестилізуємі компоненти із зовнішніх бібліотек в VCL застосунках, що використовуть стиль.

Поліпшена кроссплатформеність

- Додана підтримка Metal Driver GPU для macOS і iOS.
- Крім підтримки останнього iOS SDK, в RAD Studio 10.4 розроблювачі можуть задовольнити нові вимоги Apple до набору стартових екранів.
- Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.
- Користувачам редакцій Enterprise або Architect доступна повна інтеграція Fmxlinux з IDE для створення клієнтських застосунків Linux з GUI.
- Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API.
- Реалізація компонента Media Player для macOS тепер використовує Avfoundation.

Оновлений менеджер пакетів Getit

Менеджер пакетів Getit в IDE був значно вдосконалений.

Дати випуску релізів пакетів тепер видні, і можливе сортування списку по цих датах; відбір тільки встановлених пакетів, контенту, доступного тільки при наявності підписки, багато чого іншого.

Універсальний інсталятор для установки Online і Offline

В 10.4 включений новий універсальний інсталятор, який використовує технологію на базі Getit. Цей інсталятор підтримує як online, так і offline (з ISO) варіанти установки.

Тепер обоє варіанта установки дозволяють вам указати початковий набір можливостей RAD Studio для установки, наприклад, свою комбінацію мов програмування й цільових платформ, мов інтерфейсу, і додавати до нього або видаляти непотрібне в будь-який момент.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

В процесі розробки випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи кібербезпеки контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які

забезпечать впровадження системи кібербезпеки в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Комп'ютерна стеганографія

Комп'ютерна стеганографія – напрямок класичної стеганографії, засноване на особливостях комп'ютерної платформи.

Приклади – стеганографічна файлова система StegFS для Linux, приховання даних у невикористовуваних областях форматів файлів, підміна символів у назвах файлів, текстова стеганографія й т.д.

Приведемо деякі приклади:

– Використання зарезервованих полів комп'ютерних форматів файлів – суть методу полягає в тому, що частина поля розширень, не заповнена інформацією про розширення, за замовчуванням заповнюється нулями. Відповідно ми можемо використовувати цю «нульову» частину для запису своїх даних. Недоліком цього методу є низький ступінь скритності й малий обсяг переданої інформації.

– Метод приховання інформації в невикористовуваних місцях гнучких дисків – при використанні цього методу інформація записується в невикористовувані частини диска, приміром, на нульову доріжку. Недоліки: маленька продуктивність, передача невеликих по обсязі повідомлень.

– Метод використання особливих властивостей полів форматів, які не відображаються на екрані – цей метод заснований на спеціальних «невидимих» полях для одержання виносок, покажчиків. Приміром, написання чорним шрифтом на чорному тлі. Недоліки: маленька продуктивність, невеликий обсяг переданої інформації.

– Використання особливостей файлових систем – при зберіганні на жорсткому диску файл завжди (не вважаючи деяких ФС, наприклад, ReiserFS)

займає ціле число кластерів (мінімальних адресуємих обсягів інформації). Приміром, у раніше широко використовуваній файловій системі FAT32 (використовувалася в Windows 98/Me/2000) стандартний розмір кластера – 4 Кб. Відповідно для зберігання 1 Кб інформації на диску виділяється 4 Кб інформації, з яких 1Кб потрібний для зберігання файлу, що зберігається, а інші 3 ні на що не використовуються – відповідно їх можна використовувати для зберігання інформації. Недолік даного методу: легкість виявлення.

Цифрова стеганографія

Цифрова стеганографія – напрямок класичної стеганографії, заснований на прихованні або впровадженні додаткової інформації в цифрові об'єкти, викликаючи при цьому деякого перекручування цих об'єктів. Але, як правило, дані об'єкти є мультимедіа-об'єктами (зображення, відео, аудіо, текстури 3D-об'єктів) і внесення перекручувань, які перебувають нижче порога чутливості середньостатистичної людини, не приводить до помітних змін цих об'єктів. Крім того, в оцифрованих об'єктах, що споконвічно мають аналогову природу, завжди є присутнім шум квантування; далі, при відтворенні цих об'єктів з'являється додатковий аналоговий шум і нелінійні перекручування апаратури, все це сприяє більшій непомітності прихованої інформації.

З рамок цифровий стеганографії вийшло найбільш затребуваний легальний напрямок – вбудовування цифрових водяних знаків (ЦВЗ) (watermarking), що є основою для систем захисту авторських прав і DRM (Digital rights management) систем. Методи цього напрямку настроєні на вбудовування схованих маркерів, стійких до різних перетворень контейнеру (атакам).

Напівтендітн і тендітні ЦВЗ використовуються в якості аналогової ЕЦП, забезпечуючи зберігання інформації про переданий підпис і спроби порушення цілісності контейнеру (каналу передачі даних).

Наприклад, розробки Digimarc у вигляді плагинів до редактора Adobe Photoshop дозволяють вмонтувати в саме зображення інформацію про автора. Однак така мітка нестійка, втім як і абсолютна їхня більшість. Програма Stirmark,

розроблювачем якої є вчений Fabien Petitcolas, з успіхом атакує подібні системи, руйнуючи стеговкладення.

Всі алгоритми вбудовування схованої інформації можна розділити на кілька підгруп:

– Працюючі із самим цифровим сигналом. Наприклад, метод LSB.

– «Упаювання» схованої інформації. У цьому випадку відбувається накладення приховуваного зображення (звуку, іноді тексту) поверх оригіналу. Часто використовується для вбудовування ЦВЗ.

– Використання особливостей форматів файлів. Сюди можна віднести запис інформації в метадані або в різні інші не використовувані зарезервовані поля файлу.

За способом вбудовування інформації стегоалгоритми можна розділити на лінійні (адditивні), нелінійні й інші. Алгоритми адditивного впровадження інформації полягають у лінійній модифікації вихідного зображення, а її добування в декодері виробляється кореляційними методами. При цьому ЦВЗ звичайно складається із зображенням-контейнером, або «вплавляється» (fusion) у нього. У нелінійних методах вбудовування інформації використовується скалярне або векторне квантування. Серед інших методів певний інтерес представляють методи, що використовують ідеї фрактального кодування зображень. До адditивних алгоритмів можна віднести:

– A17 (Cox).

– A18 (Barni).

– L18D (Lange).

– A21 (J. Kim).

– A25 (C. Podilchuk).

Метод LSB

LSB (Least Significant Bit, найменший значущий біт) – суть цього методу полягає в заміні останніх значущих бітів у контейнері (зображення, аудіо або відеозапису) на біти приховуваного повідомлення. Різниця між порожнім і

заповненим контейнеруми повинна бути не відчутна для органів сприйняття людини.

Суть методу полягає в наступному: Допустимо, є 8-бітне зображення в градаціях сірого. 00h (00000000b) позначає чорний колір, FFh (11111111b) – білий. Усього є 256 градацій (28). Також припустимо, що повідомлення складається з 1 байта – наприклад, 01101011b. При використанні 2 молодших біт в описах пікселів, нам буде потрібно 4 пікселя. Допустимо, вони чорного кольору. Тоді пікселі, що містять сховане повідомлення, будуть виглядати в такий спосіб: 00000001 0000001000000010 00000011. Тоді колір пікселів зміниться: першого – на 1/255, другого й третього – на 2/255 і четвертого – на 3/255. Такі градації, мало того що непомітні для людини, можуть взагалі не відобразитися при використанні низькоякісних пристроїв виводу.

Методи LSB є нестійкими до всіх видів атак і можуть бути використані тільки при відсутності шуму в каналі передачі даних.

Виявлення LSB-Кодованого стего здійснюється по аномальних характеристиках розподілу значень діапазону молодших бітів відрахунків цифрового сигналу.

Всі методи LSB є, як правило, аддітивними (A17, L18D).

Інші методи приховання інформації в графічних файлах орієнтовані на формати файлів із втратою, приміром, JPEG. На відміну від LSB вони більше стійкі до геометричних перетворень. Це виходить за рахунок варіювання в широкому діапазоні якості зображення, що приводить до неможливості визначення джерела зображення.

Луна-методи

Луна-методи застосовуються в цифровий аудіостеганографії й використовують нерівномірні проміжки між луна-сигналами для кодування послідовності значень. При накладенні ряду обмежень дотримується умова непомітності для людського сприйняття. Луна характеризується трьома параметрами: початковою амплітудою, ступенем загасання, затримкою. При

досягненні якогось порога між сигналом і луною вони змішуються. У цій крапці людське вухо не може вже відрізнити ці два сигнали. Наявність цієї крапки складно визначити, і вона залежить від якості вихідного запису, слухача. Найчастіше використовується затримка близько 1/1000, що цілком прийнятно для більшості записів і слухачів. Для позначення логічного нуля й одиниці використовується дві різних затримки. Вони обидві повинні бути менше, ніж поріг чутливості юшка слухача до одержуваної луни.

Луна-методи стійкі до амплітудних і частотних атак, але нестійкі до атак за часом.

Фазове кодування

Фазове кодування (phase coding, фазове кодування) – також застосовується в цифровій аудіостеганографії. Відбувається заміна вихідного звукового елемента на відносну фазу, що і є секретним повідомленням. Фаза підряд, що йдуть елементів, повинна бути додана таким чином, щоб зберегти відносну фазу між вихідними елементами. Фазове кодування є одним з найефективніших методів приховання інформації.

Метод розширеного спектра

Метод вбудовування повідомлення полягає в тому, що спеціальна випадкова послідовність вбудовується в контейнер, потім, використовуючи погоджений фільтр, дана послідовність детектується. Даний метод дозволяє вбудовувати велику кількість повідомлень у контейнер, і вони не будуть створювати перешкоди один одному. Метод запозичений із широкополосного зв'язку.

Стеганографія й цифрові водяні знаки

Цифрові водяні знаки (ЦВЗ) використовуються для захисту від копіювання, збереження авторських прав. Невидимі водяні знаки зчитуються спеціальним пристроєм, що може підтвердити або спростувати коректність.

ЦВЗ можуть містити різні дані: авторські права, ідентифікаційний номер, що управляє інформацію. Найбільш зручними для захисту за допомогою ЦВЗ є нерухливі зображення, аудіо й відео файли.

Технологія запису ідентифікаційних номерів виробників дуже схожа на ЦВЗ, але відмінність полягає в тому, що на кожний виріб записується свій індивідуальний номер (так звані «відбитки пальців»), по якому можна обчислити подальшу долю виробу. Невидиме вбудовування заголовків іноді використовується, приміром, для підписів медичних знімків, нанесення шляху на карту й т.п. Швидше за все, цей єдиний напрямок стеганографії, де немає порушника в явному виді.

Основні вимоги, пропоновані до водяних знаків: надійність і стійкість до перекручувань, непомітності, робастність до обробки сигналів (робастность – здатність системи до відновлення після впливу на неї зовнішніх/внутрішніх перекручувань, у тому числі навмисних). ЦВЗ мають невеликий обсяг, але для виконання зазначених вище вимог, при їхньому вбудовуванні використовуються більше складні методи, чим для вбудовування звичайних заголовків або повідомлень. Такі завдання виконують спеціальні стегосистеми.

Перед приміщенням ЦВЗ у контейнер, водяний знак потрібно перетворити до підходящого виду. Приміром, якщо як контейнер використовується зображення, те й ЦВЗ повинні бути представлені як двовимірний бітовий масив.

Для підвищення стійкості до перекручувань часто застосовують завадостійке кодування або використовують широкополосні сигнали. Початкову обробку схованого повідомлення робить прекодер. Важлива попередня обробка ЦВЗ – обчислення його узагальненого Фур'є-перетворення. Це підвищує завадостійкість. Первинну обробку часто роблять із використанням ключа – для підвищення таємності. Потім водяний знак «укладається» у контейнер (наприклад, шляхом зміни молодших значущий біт). Тут використовуються особливості сприйняття зображень людиною. Широко відомо, що зображення мають величезну психовізуальну надмірність. Очі людини подібні до

низькочастотного фільтра, що пропускає дрібні елементи зображення. Найменш помітні перекручування у високочастотній області зображень. Впровадження ЦВЗ також повинне враховувати властивості сприйняття людини.

У багатьох стегосистемах для запису й зчитування ЦВЗ використовується ключ. Він може призначатися для обмеженого кола користувачів або ж бути секретним. Наприклад, ключ потрібний в DVD-плейерах для можливості прочитання ними ЦВЗ, які містяться, на дисках. Як відомо, не існує таких стегосистем, у яких би при зчитуванні водяного знака була потрібна інша інформація, ніж при його записі. У стегодетекторі відбувається виявлення ЦВЗ у захищеному їм файлі, що, можливо, міг бути змінений. Ці зміни можуть бути пов'язані із впливами помилок у каналі зв'язку, або навмисними перешкодами. У більшості моделей стегосистем сигнал-контейнер можна розглянути як адді тивний шум. При цьому завдання виявлення й зчитування стегоповідомлення вже не представляє складності, але не враховує двох факторів: невипадковості сигналу контейнеру й запитів по збереженню його якості. Облік цих параметрів дозволить будувати більше якісні стегосистеми. Для виявлення факту існування водяного знака і його зчитувань використовуються спеціальні пристрої – стегодетектори. Для винесення рішення про наявність або відсутність водяного знака використовують, приміром, відстань за Хеммінгом, взаємокорреляцію між отриманим сигналом і його оригіналом. У випадку відсутності вихідного сигналу в справу вступають більше витончені статистичні методи, які засновані на побудові моделей досліджуваного класу сигналів.

Зведена характеристика методів стеганографічного захисту

У цей час методи комп'ютерної стеганографії розвиваються по двох основних напрямках:

– Методи, засновані на використанні спеціальних властивостей комп'ютерних форматів;

– Методи, засновані на надмірності аудіо й візуальної інформації.

Таблиця 3.1 – Порівняльні характеристики стеганографічних методів

Стеганографічні методи	Коротка характеристика методів	Недоліки	Переваги
1. Методи використання спеціальних властивостей комп'ютерних форматів даних			
1.1. Методи використання зарезервованих для розширення полів комп'ютерних форматів даних	Поля розширення є в багатьох мультимедійних форматах, вони заповнюються нульовою інформацією й не враховуються програмою	Низький ступінь скритності, передача невеликих обмежених обсягів інформації	Простота використання
1.2. Методи спеціального форматування текстових файлів:			
1.2.1. Методи використання відомого зсуву слів, пропозицій, абзаців	Методи засновані на зміні положення рядків і розміщення слів у пропозиції, що забезпечується вставкою додаткових пробілів між словами	1. Слабка робастність методу, передача невеликих обсягів інформації 2. Низький ступінь скритності	Простота використання. Є опубліковане програмне забезпечення реалізації даного методу

Продовження таблиці 3.1

Стеганографічні методи	Коротка характеристика методів	Недоліки	Переваги
1.2.2. Методи вибору певних позицій букв (нульовий шифр)	Акровірш – окремий випадок цього методу (наприклад, початкові букви кожного рядка утворять повідомлення)	1. Слабка робастність методу, передача невеликих обсягів інформації 2. Низький ступінь скритності	Простота використання. Є опубліковане програмне забезпечення реалізації даного методу
1.2.3. Методи використання спеціальних властивостей полів форматів, не відображуваних на екрані	Методи засновані на використанні спеціальних "невидимих", схованих полів для організації виносок і посилань (наприклад, використання чорного шрифту на чорному тлі)		
1.3. Методи приховання в невикористовуваних місцях гнучких дисків	Інформація записується у звичайно невикористовуваних місцях ГМД (наприклад, у нульовій доріжці)	1. Слабка продуктивність методу, передача невеликих обсягів інформації 2. Низький ступінь скритності	Простота використання. Є опубліковане програмне забезпечення реалізації даного методу

Продовження таблиці 3.1

Стеганографічні методи	Коротка характеристика методів	Недоліки	Переваги
1.4. Методи використання функцій, що імітують (mimic-function)	Метод заснований на генерації текстів і є узагальненням акровірша. Для таємного повідомлення генерується осмислений текст, що приховує саме повідомлення	1. Слабка продуктивність методу, передача невеликих обсягів інформації 2. Низький ступінь скритності	Результуючий текст не є підозрілим для систем моніторингу мережі
1.5. Методи видалення ідентифікуючий файл заголовка	Приховуване повідомлення шифрується й у результату віддаляється ідентифікуючий заголовок, залишаючи тільки шифровані дані. Одержувач заздалегідь знає про передачу повідомлення й має відсутній заголовок	Проблема приховання вирішується тільки частково. Необхідно заздалегідь передати частина інформації одержувачеві	Простота реалізації. Багато засобів (White Noise Storm, S-Tools), забезпечують реалізацію цього методу

Продовження таблиці 3.1

Стеганографічні методи	Коротка характеристика методів	Недоліки	Переваги
2. Методи використання надмірності аудіо й візуальної інформації			
2.1. Методи використання надмірності цифрові фотографії, цифрового звуку й цифрового відео	Молодші розряди цифрових відрахунків містять дуже мало корисної інформації. Їхнє заповнення додатковою інформацією практично не впливає на якість сприйняття, що й дає можливість приховання конфіденційної інформації	За рахунок введення додаткової інформації спотворюються статистичні характеристики цифрових потоків. Для зниження компрометуючих ознак потрібна корекція статистичних характеристик	Можливість схованої передачі великого обсягу інформації. Можливість захисту авторського права, схованого зображення товарної марки, реєстраційних номерів і т.п.

Як видно з таблиці 3.1, перший напрямок заснований на використанні спеціальних властивостей комп'ютерних форматів подання даних, а не на надмірності самих даних. Спеціальні властивості форматів вибираються з урахуванням захисту приховуваного повідомлення від безпосереднього прослуховування, перегляду або прочитання. На підставі аналізу матеріалів таблиці 3.1 можна зробити вивід, що основним напрямком комп'ютерної стеганографії є використання надмірності аудіо й візуальної інформації.

Цифрові фотографії, цифрова музика, цифрове відео – представляються матрицями чисел, які кодують інтенсивність у дискретні моменти в просторі й/або в часі.

Цифрова фотографія – це матриця чисел, що представляють інтенсивність світла в певний момент часу.

Цифровий звук – це матриця чисел, що представляє інтенсивність звукового сигналу в послідовно, що йдуть моменти, часу. Всі ці числа не точні, тому що не точні пристрої оцифровки аналогових сигналів, є шуми квантування. Молодші розряди цифрових відрахунків містять дуже мало корисної інформації про поточні параметри звуку й візуальнного образа. Їхнє заповнення відчутне не впливає на якість сприйняття, що й дає можливість для приховання додаткової інформації.

Графічні кольорові файли зі схемою змішання RGB кодують кожну крапку рисунка трьома байтами. Кожна така крапка складається з аддітивних складових: червоного, зеленого, синього. Зміна кожного із трьох найменш значимий біт приводить до зміни менш 1% інтенсивності даної крапки. Це дозволяє приховувати в стандартній графічній картинці обсягом 800 Кбайт близько 100 Кбайт інформації, що не помітно при перегляді зображення.

Інший приклад. Тільки одна секунда оцифрованного звуку із частотою дискретизації 44100 Гц і рівнем відліку 8 біт у стерео режимі дозволяє сховати за рахунок заміни найменш значимих молодших розрядів на приховуване повідомлення близько 10 Кбайт інформації. При цьому зміна значень відрахунків становить менш 1%. Така зміна практично не виявляється при прослуховуванні файлу більшістю людей.

Реалізація алгоритму стеганографічної захисту

Особливості реалізації алгоритму

Для реалізації виберемо два алгоритми стеганографії – перший алгоритм LSB, а другий – алгоритм, заснований на використанні резервних полів.

Алгоритм заміни молодших значущих бітів буде робити зміну бітів безпосередньо самого рисунка, при цьому замінятися будуть три молодших біти.

Так як сховане повідомлення може бути дуже маленьким, а зображення, у яке буде міститися даний текст більшим, те необхідно також помістити інформацію про розмір повідомлення. Для цього нехай будуть використовуватися дві останніх пікселя картинки.

Зберегти змінене зображення можна тільки у форматі BMP, тому що збереження в інших форматах стискає тим або іншим способом зображення, що неминуче спричинить ушкодження повідомлення.

Текст повідомлення буде зберігатися в багаторядковому поле, отже, необхідно передати всі знаки переходу на новий рядок. Для того щоб уникнути проблем при роботі зі знаками переходу на новий рядок будемо використовувати потоки, які дозволять працювати з текстовими даними як з набором біт.

Програма як вихідне зображення може використовувати зображення формату JPEG або BMP, але кінцеве зображення буде зберігатися винятково в BMP форматі.

Для використання резервних полів файлу JPEG скористаємося набором сторонніх бібліотек CCR.Exif-1.1.2. Повідомлення буде зберігатися у відкритому виді в полі коментарів до файлу – така примітивна вставка може обдурити як машину, що шукає потрібну інформацію з певних критеріїв, так і людини, що з великою часткою ймовірності, при поверхневому перегляді файлу не зверне увагу на зміст поля коментарів.

Написана програма реалізує стеганографічні алгоритми, які дозволяють утаємничити сам факт передачі інформації, хоча інформація передається у відкритому виді.

Існуючі сьогодні стеганографічні алгоритми, які мають найбільше поширенні – це алгоритми вбудовування схованої інформації, які можна розділити на кілька підгруп:

– Працюючі із самим цифровим сигналом. Наприклад, метод LSB, що був реалізований у програмі.

– «Упаювання» схованої інформації. У цьому випадку відбувається накладення приховуваного зображення (звуку, іноді тексту) поверх оригіналу. Часто використовується для вбудовування ЦВЗ.

– Використання особливостей форматів файлів.

Хоча стеганографічні методи захисту інформації дозволяють приховувати сам факт передачі інформації, але використання тільки стеганографії не дозволяє захищати інформацію на належному рівні. Для підвищення захищеності переданої інформації необхідно використовувати додаткові заходи захисту, такі як використання криптографічних протоколів. Тоді у випадку перехоплення повідомлення й виявлення факту передачі супротивникові знадобиться, час для розкриття повідомлення, який може виявитися досить для того щоб до моменту розкриття повідомлення інформація втратила всяку актуальність.

3.2 Розробка структурної схеми

Структурна схема розробленої системи зображена на рисунку 3.1. На ній аналізуються методи, призначені для забезпечення безпеки передачі даних. Ці методи, з огляду на природні неточності пристроїв оцифровки й надмірність аналогового відео– або аудіо-сигналу, дозволяють приховувати повідомлення в комп'ютерних файлах (контейнерах), що дає можливість говорити про становлення нового швидко, що розвивається напрямку, у сфері захисту інформації – комп'ютерної стеганографії (КС), що займається питаннями реалізації стегосистем з використанням комп'ютерної техніки. Основними вихідними положеннями сучасної КС є наступні:

– методи приховання повинні забезпечувати незмінність і цілісність файлу;

– супротивникові повністю відомі можливі стеганографічні методи;

– безпека методів ґрунтується на збереженні стеганографічним перетворенням основних властивостей відкрито переданого файлу при внесенні в нього секретного повідомлення й деякої невідомої супротивникові інформації – ключа;

– якщо факт приховання повідомлення став відомий супротивникові, добування самого секретного повідомлення представляє складне обчислювальне завдання.

Існує два основних напрямки використання КС. У першому випадку секретні повідомлення вбудовуються в цифрові дані, які, як правило, мають аналогову природу – мова, зображення, аудіо– і відеозапису. У другому – конфіденційна інформація розміщається в заголовках файлів різних форматів і в текстових повідомленнях. Переважна більшість поточних досліджень у сфері стеганографії, так чи інакше, пов'язано саме із цифровою обробкою сигналів, що дозволяє говорити про цифровий стеганографії (ЦС), як про науку, про непомітне й надійне приховання одних бітових послідовностей в інші, що мають аналогову природу. У цьому визначенні втримуються дві головних вимоги до стеганографічного перетворення: непомітність і надійність, тобто стійкість до різного роду перекручуванням.

У цей час можна виділити чотири тісно зв'язаних між собою й маючих те саме коріння напрямку додатків ЦС:

– вбудовування інформації з метою її схованої передачі;

– вбудовування цифрових водяних знаків (ЦВЗ);

– вбудовування ідентифікаційних номерів (fingerprinting);

– вбудовування заголовків (captioning).

Однієї з найважливіших проблем стеганографії, є проблема стійкості стеганографічних систем. Кожна із зазначених вище областей застосування стеганографії вимагає певного співвідношення між стійкістю убудованого повідомлення до зовнішніх впливів і розміром убудованого повідомлення. Для більшості сучасних методів, які використовуються для приховання повідомлень у

файлах цифрового формату, має місце залежність надійності системи від обсягу даних, що вбудовуються.

Збільшення обсягу даних, що вбудовуються, значно знижує надійність системи. Таким чином, є проблема ухвалення оптимального рішення при виборі між кількістю (обсягом) приховуваних даних і ступенем стійкості (утаємничення) до можливої модифікації (аналізу) сигналу – контейнеру.

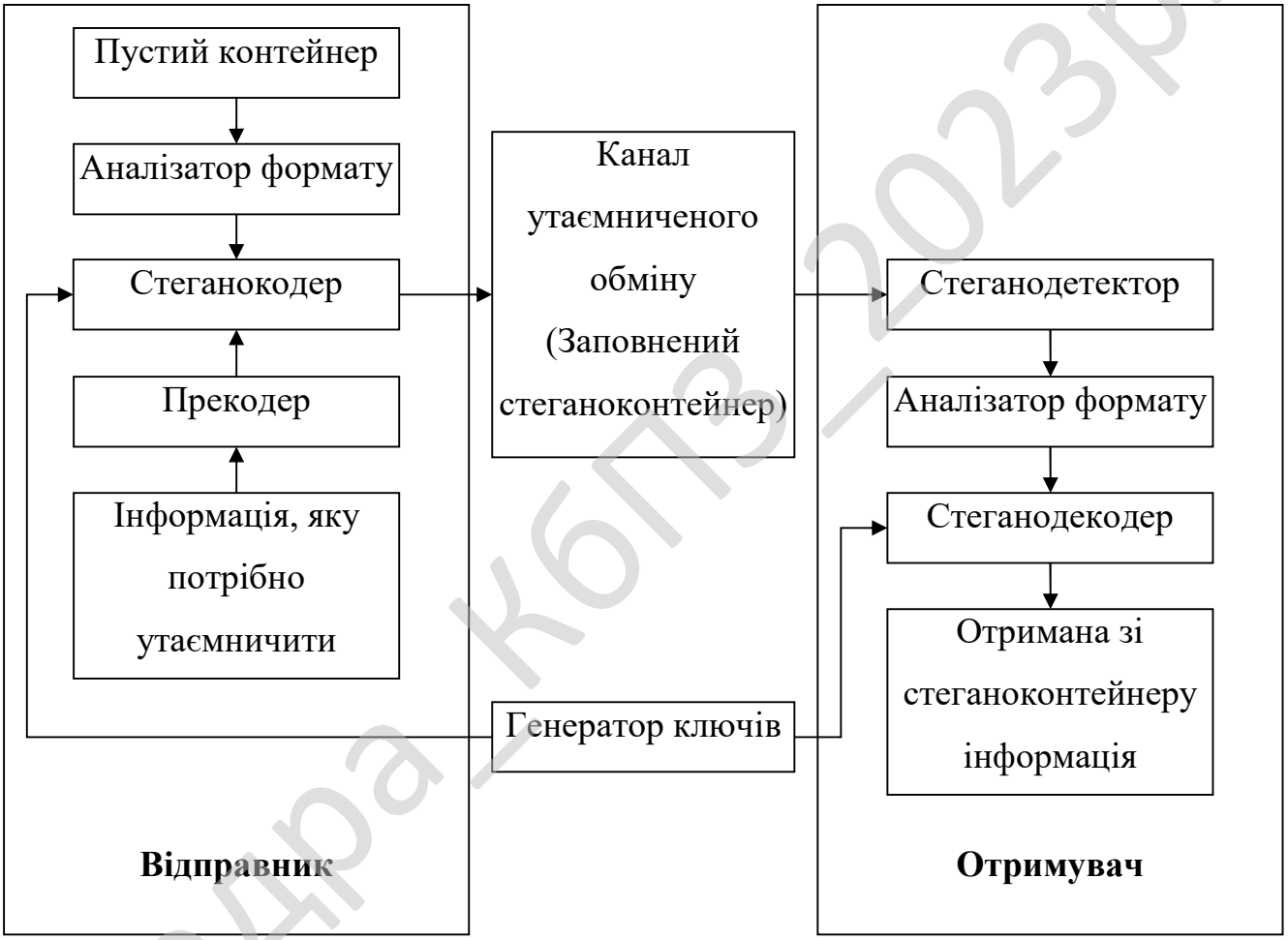

Рисунок 3.1 – Структурна схема системи

Як би не відрізнялися напрямки використання стеганографії, висунуті при цьому вимоги, багато в чому, залишаються незмінними. Так, властивості контейнеру повинні бути модифіковані настільки, щоб зміни неможливо було виявити при візуальному контролі. Ця вимога визначає якість приховання

впроваджуваного повідомлення: забезпечення безперешкодного проходження стегоповідомлення по каналі зв'язку так, щоб воно ніяким образом не могло привернути увагу супротивника.

Математичну модель стеганосистеми можна представити у вигляді двох залежностей:

$$E : C \times M \rightarrow S, \quad (3.1)$$

$$D: S \rightarrow M, \quad (3.2)$$

де S – безліч контейнерів – результатів.

C – безліч контейнерів – оригіналів.

M – безліч секретних повідомлень.

E – алгоритм прямого стеганографічного перетворення.

D – алгоритм зворотного стеганографічного перетворення

З огляду на велику розмаїтість стеганографічних систем, доцільно звести їх до наступних чотирьох типів:

– безключові стеганосистеми;

– стеганосистеми із секретним ключем;

– стеганосистеми з відкритим ключем;

– змішані стеганосистеми.

Різні види атак і основні етапи взлому стеганосистеми:

– виявлення факту присутності схованої інформації;

– добування схованого повідомлення;

– видозміна (модифікація) схованої інформації;

– заборона на виконання будь-якого пересилання інформації, у тому числі схованої.

Перші два етапи відносяться до пасивних атак на стеганосистему, а останні – до активного (або зловмисним) атакам.

Стеганосистема вважається зламаною, якщо порушникові вдалося, принаймні, довести існування схованого повідомлення в перехопленому контейнері. Передбачається, що порушник здатний здійснювати будь-які типи

атак і має необмежені обчислювальні можливості. Якщо йому не вдається підтвердити гіпотезу про те, що в контейнері сховане секретне повідомлення, то стеганографічна система вважається стійкою.

Стеганографія являє собою науку про методи приховання інформації в контейнерах без порушення їхньої природності. Найпоширенішими типами контейнерів на даний момент є растрові графічні зображення, представлені в цифровому виді різних форматів, а також відеопослідовності. Це пояснюється тим, що подібні контейнери вже за технологією одержання мають шумову складову, що маскує повідомлення, що вбудовується.

Переважна більшість методів КС базуються на двох ключових принципах:

– файли, які не вимагають абсолютної точності (наприклад, файли із зображенням, звуковою інформацією й т.д.), можуть бути видозмінені (звичайно, до певного ступеня) без втрати своєї функціональності;

– органи почуттів людини нездатні надійно розрізняти незначні зміни в модифікованих у такий спосіб файлах, або відсутній спеціальний інструментарій, що був би здатний виконувати дане завдання.

3.3 Розробка функціональної схеми

Функціональна схема розробленої системи зображена на рисунку 3.2. Система включає дві процедури:

– процедуру вбудовування конфіденційної інформації в стеганоконтейнер;

– процедуру вилучення конфіденційної інформації із стеганоконтейнера.

У системі задаються ключі, за допомогою, яких вбудовуються дані у стеганоконтейнер.

Щоб стегоключ неможливо було взнати зловмиснику, він зашифровується алгоритмом AES, та передається отримувачу повідомлення.

Так, як ключ кодується за допомогою алгоритму AES, то наведемо принцип роботи цього алгоритму.

Для захисту розробленого програмного забезпечення запропоновано використати фіналіста конкурсу AES – шифр Rijndael. Він є нетрадиційним блоковим шифром, оскільки не використовує мережу Фейштеля для криптоперетворень. Алгоритм представляє кожний блок кодуємих даних у вигляді двовимірного масиву байт розміром 4x4, 4x6 або 4x8 залежно від установленої довжини блоку. Далі на відповідних етапах перетворення відбуваються або над незалежними стовпцями, або над незалежними рядками, або взагалі над окремими байтами в таблиці.

Всі перетворення в шифрі мають строге математичне обґрунтування. Сама структура й послідовність операцій дозволяють виконувати даний алгоритм ефективно як на 16-бітних так і на 64-бітних процесорах. У структурі алгоритму закладена можливість паралельного виконання деяких операцій, що на багатопроцессорних робочих станціях може ще підняти швидкість шифрування в 4 рази.

Алгоритм складається з деякої кількості раундів (від 10 до 14 – це залежить від розміру блоку й довжини ключа), у яких послідовно виконуються наступні операції :

ByteSub – Таблична підстановка 8x8 біт (рисунок 3.2).

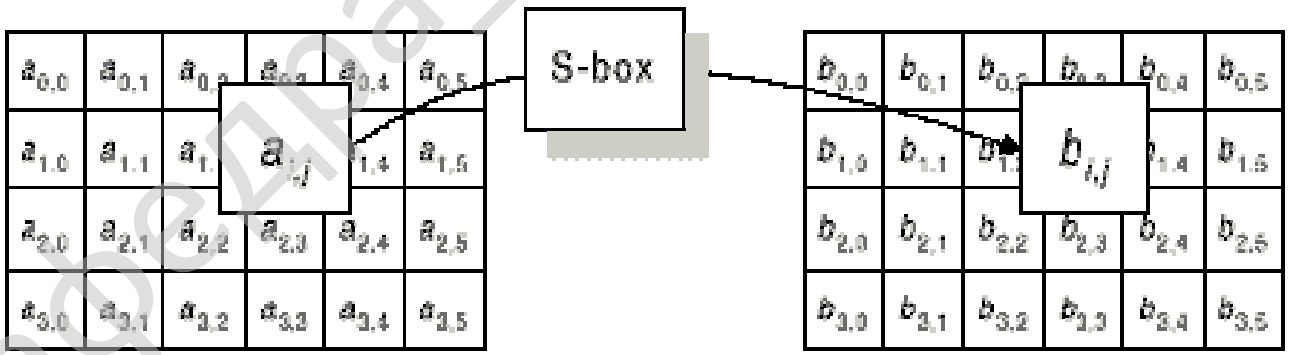

Рисунок 3.2 – Таблична підстановка 8x8 біт

ShiftRow – зрушення рядків у двовимірному масиві на різні зсуви (рисунок 3.3).

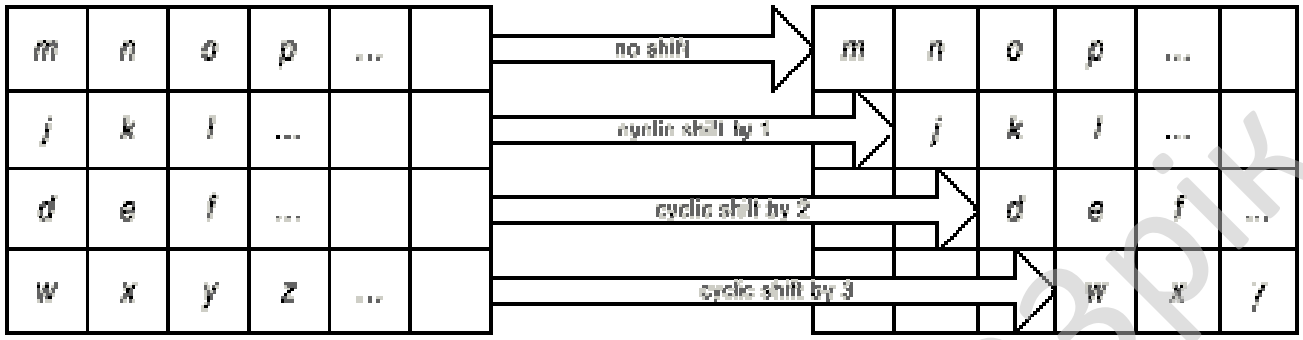

Рисунок 3.3 – Зрушення рядків у двовимірному масиві на різні зсуви

MixColumn – математичне перетворення, що перемішує дані усередині стовпця (рисунок 3.4).

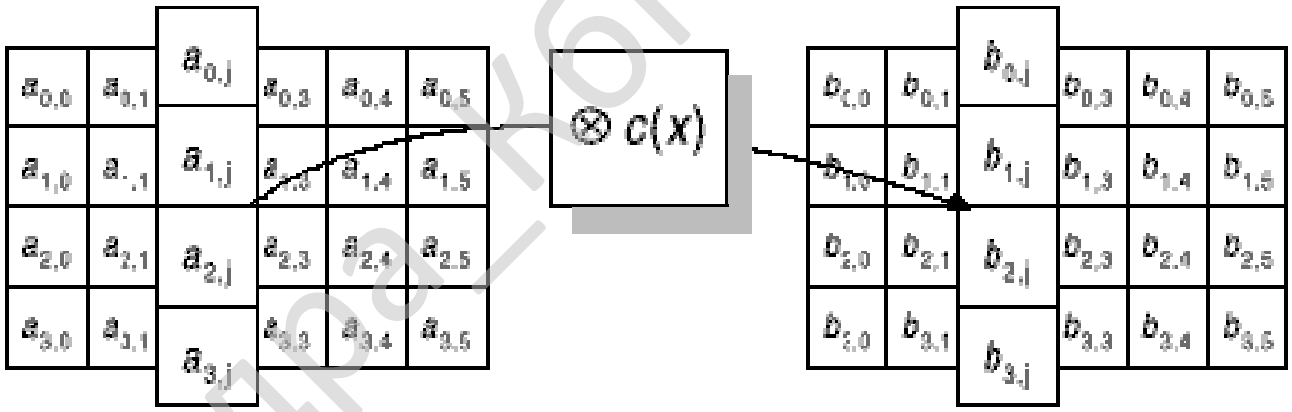

Рисунок 3.4 – Математичне перетворення, що перемішує дані усередині стовпця

AddRoundKey – додавання матеріалу ключа операцією XOR (рисунок 3.5).

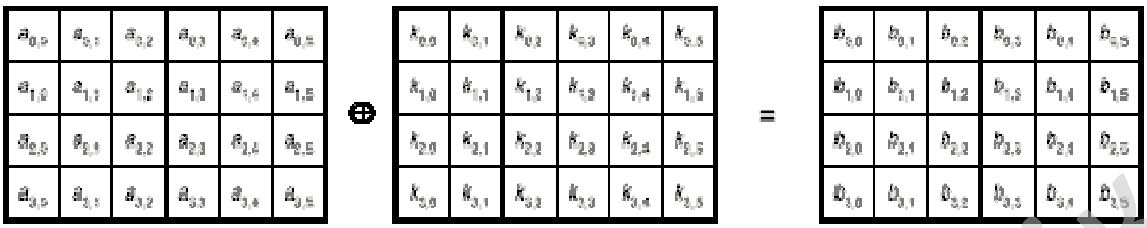

Рисунок 3.5 – Додавання матеріалу ключа операцією XOR

В останньому раунді операція перемішування стовпців відсутня, що робить всю послідовність операцій симетричною.

Робота системи відбувається наступним чином.

Спершу опишемо процедуру вбудовування конфіденційної інформації в стеганоконтейнер.

Для цього береться повідомлення, над яким відбувається операція кодування у вибраному зображені, яке є контейнером.

Контейнер – це те зображення, куди буде приховано записано повідомлення.

Принцип кодування наступний:

1. Обчислюється контрольна сума пароля

2. Обчислюється контрольний добуток пароля

3. Всі закодовані дані представляються як масив байтів

4. Від кожного байта даних віднімається байт контрольної суми пароля

5. З результатом попереднього обчислення робиться XOR з байтом контрольного добутку пароля

6. До результату попереднього обчислення додається код відповідного символу з рядка пароля.

7. Як тільки рядок пароля закінчується знову переходимо на його початок.

Рисунок 3.6 – Функціональна схема системи

Для реалізації цього методу відбувається генерація стегоключа, за допомогою якого повідомлення буде зашифроване.

Для більш високої надійності передачі даних, повідомлення кодується за допомогою перешкодостійкого кодеку Ріда-Соломона.

Після цього відбувається передача закодованого зображення з прихованим повідомленням, по каналам зв’язку.

На приймальній стороні отримують зображення й реалізують процедуру вилучення конфіденційної інформації із стеганоконтейнера.

Це відбувається наступним чином.

Спершу отримане повідомлення перевіряють на наявність помилок, за рахунок застосування кодеку Ріда-Соломона.

Якщо є помилки, то вони виправляються, за рахунок властивостей цього кодеку з виявлення та виправлення помилок.

Після цього відбувається дешифрування ключа. За допомогою отриманого ключа відбувається створення файлу з повідомлення та очищення контейнера.

Після цього на стороні отримувача є інформація, яка була прихована у зображені, та саме зображення, яке може служити контейнером для наступної інформації, яку треба приховано передати.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.7. Першим процесом, який запускається у системі, є процес виведення головного вікна програми, який взаємодіє з процесом відкриття файлу з зображенням-контейнером. Останній процес взаємодіє з наступними процесами:

– Процес читання стего.

– Процес приховування стего.

Процес читання стего взаємодіє з наступними процесами:

– Процес обнуління бітів, що містять стего.

– Процес виведення на екран стего.

Процес виведення на екран стего взаємодіє з наступними процесами:

– Виведення текстового повідомлення.

– Виведення прихованого зображення.

Рисунок 3.7 – Діаграма взаємодії процесів

Процес приховування стего взаємодіє з наступними процесами:

– Процес створення стегоключа для приховування даних.

– Процес приховування текстового повідомлення.

– Процес приховування зображення / цифрового водяного знаку.

Процес створення стегоключа для приховування даних взаємодіє з наступними процесами:

– Процес введення стегоключа.

– Процес генерації стегоключа, який, у свою чергу, взаємодіє з процесом збереження стегоключа у текстовий файл.

Процес приховування текстового повідомлення взаємодіє з наступними процесами:

– Процес введення тексту.

– Процес відкриття текстового файлу.

– Процес шифрування тексту для більшої надійності.

Процес приховування зображення / цифрового водяного знаку взаємодіє з процесом відкриття зображення для приховування, який, у свою чергу, взаємодіє з процесом вибору якості зображення для приховування.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

На рисунку 4.1 зображена блок-схема основної програми. Як видно з рисунку робота програми складається з наступної послідовності кроків:

Крок 1. Виведення основного вікна програми на екран.

Крок 2. Відкриття контейнеру-зображення та виведення його на екран.

Крок 3. Якщо користувач бажає приховати текстове повідомлення, він вводить вручну або з текстового файлу стегоповідомлення.

Крок 4. Користувач вводить стегоключ або генерує його автоматично.

Крок 5. За необхідності створення додаткового рівня захисту, відбувається шифрування текстового повідомлення.

Крок 6. Викликається підпрограма приховання стегоповідомлення.

Крок 7. Якщо користувач бажає приховати графічне повідомлення, він відкриває текстовий файл з зображенням для приховання.

Крок 8. Користувач вводить стегоключ або генерує його автоматично.

Крок 9. Викликається підпрограма приховання стегоповідомлення.

Крок 10. Якщо користувач бажає здійснити читання прихованого стегоповідомлення, то він відкриває заповнене зображення контейнер.

Крок 11. Користувач вводить стегоключ вручну або з текстового файлу.

Крок 12. Викликається підпрограма читання стегоповідомлення.

Крок 13. Вилучене стегоповідомлення виводиться на екран. Якщо це текст – у текстовому полі, якщо зображення у панелі виведення зображень.

Крок 14. Якщо користувач бажає створити новий стегоключ, то він здійснює автоматичну генерацію ключа, або вводить його вручну, після чого записує у текстовий файл для подальшого використання.

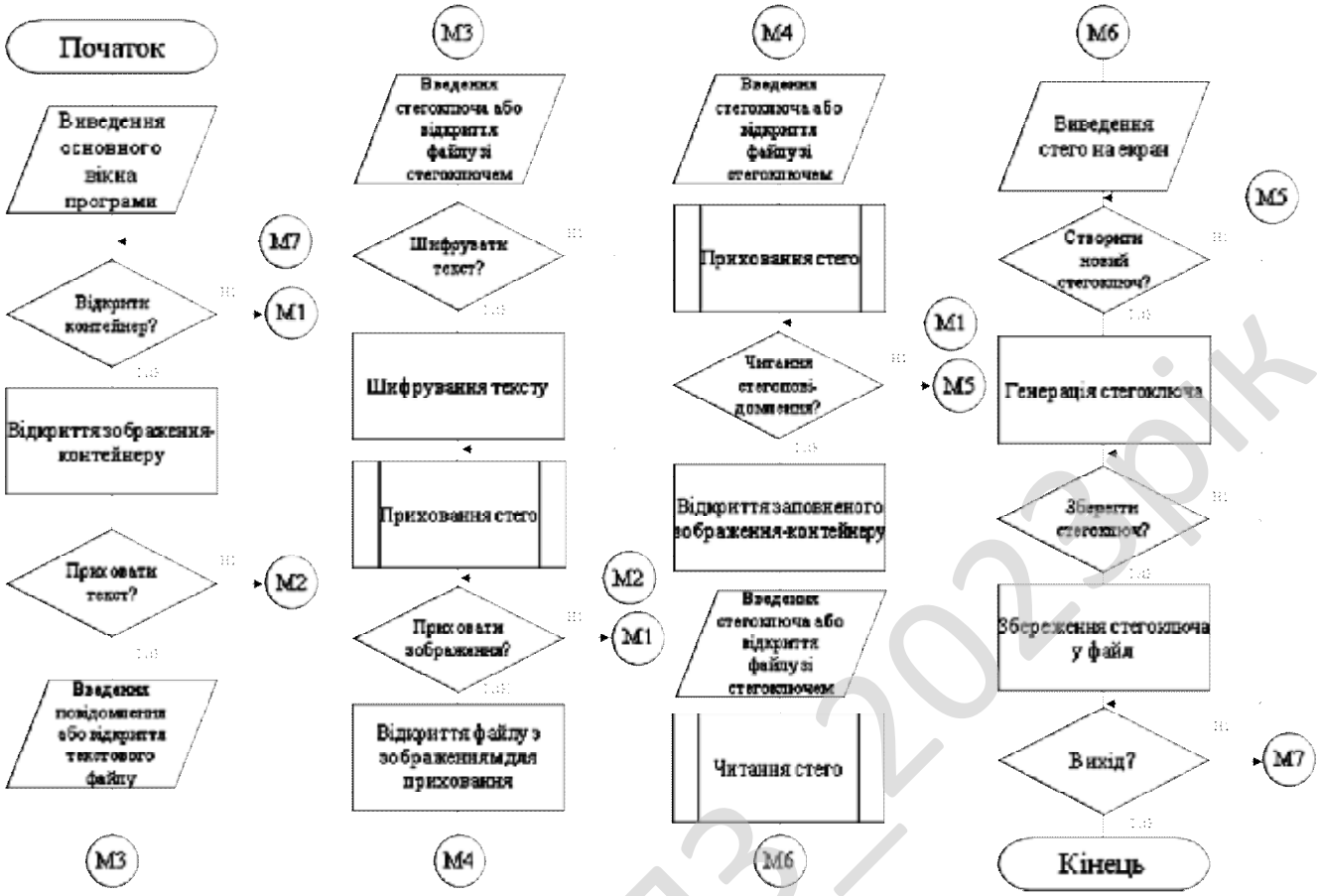

Рисунок 4.1 – Блок-схема основної програми

Для реалізації стегосистеми було взято за основу та модифіковано два стеганографічні алгоритми: Метод Куттера-Джордана-Боссена та Метод Дармстедтера-Делейгла-Квисквотера-Макка. Розглянемо дані методи детальніше.

Метод Куттера-Джордана-Боссена

Куттер, Джордан і Боссен запропонували алгоритм вбудовування інформації в канал синього кольору зображення, що має RGB-кодування, оскільки до синього кольору людський зір є найменш чутливим.

Розглянемо алгоритм передачі одного біта секретної інформації в запропонованому методі. Нехай M_i – біт, що підлягає вбудовуванню, $C = \{R, G, B\}$ – зображення-контейнер, $p = (x, y)$ – псевдовипадковий піксель контейнера, у який буде виконуватися вбудовування.

Секретний біт M_i вбудовується в канал синього кольору шляхом

модифікації яскравості $\lambda_{x,y} = 0{,}29890R_{x,y} + 0{,}58662G_{x,y} + 0{,}11448B_{x,y}$:

$$X = B_{x,y} + (2m_i - 1)\upsilon\lambda_{x,y}, \quad (4.1)$$

де υ – константа, яка визначає енергію сигналу, що вбудовується. Її величина залежить від призначення стеганосистеми. Чим більше υ, тим вище стійкість вбудованої інформації до перекручувань, однак і тем сильніше її помітність.

Одержувач витягає біт, не маючи первинного зображення, тобто, «наосліп». Для цього виконується пророкування значення первинного, не модифікованого пікселя на основі значень сусідніх пікселів. Для одержання оцінки пікселя запропоновано використовувати значення декількох пікселів розміщених у тому ж стовпці й у тому ж рядку масиву графічного контейнера. Автори використовували хрест пікселів розміром 7×7. Оцінка виходить у вигляді:

$$\tilde{B}^*_{x,y} = \frac{1}{4\delta}\left[\sum_{t=-\delta}^{+\delta} B^*_{x+i,y} + \sum_{t=-\delta}^{+\delta} B^*_{x,y+j} - 2B^*_{x,y}\right], \quad (4.2)$$

де σ – кількість пікселів зверху (знизу, ліворуч, праворуч) від оцінюваного пікселя (у випадку хреста 7x7 $\sigma = 3$).

При добуванні встроєного біта обчислюється різниця δ між поточним і прогнозованим значеннями інтенсивності пікселя p=(x,y):

$$\delta = B^*_{x,y} - \tilde{B}_{x,y} \quad (4.3)$$

Знак δ буде означати вбудований біт:

– якщо $\delta < 0$, то $M_i = 0$;

– якщо $\delta > 0$, то $M_i = 1$.

Функції вбудовування й добування в даному методі не симетричні, тобто, функція добування не є зворотною функцією вбудовування. Хоча, як вказують автори методу, правильне розпізнання біта повідомлення у випадку застосування описаних вище процедур є високоймовірним, однак не стовідсотковим. Для зменшення ймовірності помилок добування було запропоновано в процесі вбудовування кожний біт повторювати кілька разів (багаторазове вбудовування). Оскільки при цьому кожний біт був повторений τ разів, то виходить τ оцінок одного біта повідомлення. Секретний біт витягається за результатами

усереднення різниці між реальним і оціненим значеннями інтенсивності пікселя в отриманому контейнері:

$$\delta = \tau^{-1} \sum_{i=1}^{\tau} \left[B^{*}_{x,y} - \tilde{B}^{*}_{x,y} \right] \quad (4.4)$$

Як і в попередньому випадку, знак усередненої різниці δ буде визначати значення вбудованого біта. Даний алгоритм стійкий до багатьох відомих видів атак: низькочастотної фільтрації зображення, його компресії відповідно до алгоритму JPEG, обрізання країв.

Метод Дармстедтера-Делейгла-Квісквотера-Макка

Нетрадиційний блоковий метод вбудовування в просторову область контейнера запропонували Дармстедтер, Делейгл, Квісквотер і Макк. Розроблений ними метод базується на елементарному перцепціонному сприйнятті й дозволяє пристосовувати вбудовування щодо поточного вмісту блоків контейнера. Перед вбудовуванням, конфіденційна інформація перетвориться у вектор двійкових даних. Кожний біт вбудовується в окремий блок. У розглянутому авторами варіанті розмірність блоків становила 8x8 пікселів. Головна причина такого вибору, мабуть, – співмірність із блоками, які використовуються при JPEG-компресії. Таким чином, дія компресії буде однаково поширюватися на кожний вбудований біт. Крім того, при цьому інформація вбудовується з надмірністю, що збільшить загальну стійкість стеганосистеми.

У загальному випадку процес вбудовування біт повідомлення виконується в чотири етапи:

– Розбивка масиву зображення-контейнера на блоки 8x8 пісксселів;

– Класифікація пікселів окремого блоку на зони із приблизно однаковими значеннями яскравості;

– Розбивка кожної зони на категорії відповідно до індивідуальної (псевдовипадкової) маски;

– Вбудовування біта залежно від співвідношення між середніми значеннями категорій кожної зони шляхом модифікації значень яскравості

кожної категорії в кожній зоні.

Для підвищення ймовірності правильного читання бітів стегоповідомлення із контейнера та збільшення стійкості алгоритму до атак було використано алгоритм завадостійкого кодування Ріда-Соломона.

Алгоритм Ріда-Соломона

Коди Ріда-Соломона – недвійкові циклічні коди, що дозволяють виправляти помилки в блоках даних. Елементами кодового вектора є не біти, а групи бітів (блоки). Дуже поширені коди Ріда-Соломона, що працюють із байтами (октетами). У даний час широко використовується в системах відновлення даних з компакт-дисків, при створенні архівів з інформацією для відновлення у випадку ушкоджень, у завадостійкому кодуванні.

Якщо говорити спрощено, то основна ідея перешкодозахисного кодування Ріда-Соломона полягає в множенні інформаційного слова, представленого у вигляді полінома D, на не незвідний полином G, відомий обом сторонам, у результаті чого виходить кодове слово C, також представлене у вигляді полінома.

Декодування здійснюється з точністю до навпаки: якщо при розподілі кодового слова C на поліном G декодер раптово одержує залишок, то він може рапортувати про помилку. Відповідно, якщо кодове слово розділилося націло, його передача завершилася успішно.

Якщо ступінь полінома G (називаного також породжуючим поліномом) перевершує степінь кодового слова щонайменше на два степені, то декодер може не тільки виявляти, але й виправляти одиночні помилки. Якщо ж перевага степеня породжую чого полінома над кодовим словом дорівнює чотирьом, то відновленню піддаються й подвійні помилки. Коротше кажучи, степінь полінома k пов'язана з максимальною кількістю помилок, що виправляються, t у такий спосіб: k = 2 * t. Отже, кодове слово повинне містити два додаткових символи на одну помилку, що виправляється. У той же час, максимальна кількість розпізнаваних помилок дорівнює t, тобто надмірність становить один символ на кожну розпізнавану помилку.

На відміну від кодів Хемінга, коди Ріда-Соломона можуть виправляти будь-яку розумну кількість помилок при цілком прийнятному рівні надмірності. Це досягається за рахунок того, що у кодах Хемінга контрольні біти контролювали лише ті інформаційні біти, що перебувають праворуч від них і ігнорували всі "лівосторонні" біти. У кодах же Ріда-Соломона контрольні біти поширюють свій вплив на всі інформаційні біти й тому зі збільшенням кількості контрольних бітів, збільшується й кількість розпізнаваних/усуваємих помилок. Саме завдяки останній обставині, власне, і викликана приголомшуюча популярність коригуючих кодів Ріда-Соломона.

Для роботи з кодами Ріда-Соломона звичайна арифметика, на жаль, не підходить і от чому. Кодування припускає обчислення за правилами дії над багаточленами, з коефіцієнтами яких треба виконувати операції додавання, віднімання, множення й ділення, причому всі ці дії не повинні супроводжуватися яким-небудь округленням проміжних результатів (навіть при розподілі!), щоб не вносити невизначеність. Причому, і проміжні, і кінцеві результати не мають права виходити за межі встановленої розрядної сітки. Як же це реалізувати при множенні. Множити інформаційне слово на породжуючий поліном зовсім не обов'язково, можна виконати наступні дії:

– додаємо до вихідного інформаційного слова D праворуч k нулів, у результаті чого в нас виходить слово довжини n = m + r і поліном $X^r * D$, де m – довжина інформаційного слова;

– ділимо отриманий поліном $X^r * D$ на породжуючий поліном G і обчислюємо залишок від ділення R, такий що: Xr * D = G * Q + R, де Q – частка, що ігноруємо через непотрібність – беремо тільки залишок;

– додаємо залишок R до інформаційного слова D, у результаті чого одержуємо кодове слово C, інформаційні біти якого зберігаються окремо від контрольних бітів. Одержаний в результаті ділення залишок – і є коригувальні коди Ріда-Соломона. Спосіб кодування, при якому інформаційні й контрольні

символи зберігаються роздільно, називається систематичним кодуванням і таке кодування досить зручне з погляду апаратної реалізації.

– Інформаційне слово + коригувальні коди можна записати так:

$$T = X^r * D + R = GQ.$$

Декодування отриманого слова T здійснюється точно так же, як у було описано вище. Якщо при діленні T (яке в дійсності є добутком G на Q) на породжуючий поліном G утворяться залишок, то слово T перекручене й, відповідно, навпаки.

Відновлення даних. Використовуючи поліном синдрому й корінь полінома локатора, можна визначити характер руйнувань кожного з перекручених символів. Зазвичай для цієї мети використовується алгоритм **Форні** (*Forney*), що складається із двох стадій: спочатку шляхом згортки полінома синдрому поліномом локатора L одержуємо деякий *проміжний поліном*, умовно позначуваний грецькою буквою W. Потім, на основі W-полінома, обчислюється *нульова позиція помилки* (*zero error location*), що у свою чергу ділиться на похідну від L-полінома. У результаті виходить бітова маска, кожний із встановлених бітів якої відповідає перекрученому біту й для відновлення кодового слова у вихідний вид, всі перекручені біти повинні бути інвертовані, що здійснюється за допомогою логічної операції XOR.

На цьому процедура декодування прийнятого кодового слова вважається закінченою. Залишається відітнути $(n - k)$ символів парності й отримане інформаційне слово готове до використання.

На рисунку 4.2 зображена блок-схема підпрограми приховування стегоповідомлення. Робота даної підпрограми складається з наступної послідовності кроків:

Крок 1. Кодування стегоповідомлення алгоритмом Ріда-Соломона.

Крок 2. Розділення контейнеру на фрагменти 8x8 пікселів.

Крок 3. Ініціалізація лічільника фрагментів та лічільника бітів стегоповідомлення.

Крок 4. Вибір пікселя з поточного фрагменту відповідно до стегоключа.

Крок 5. Вибір біту стегоповідомлення відповідно до лічильника бітів для вбудовування в контейнер.

Крок 6. Вбудовування вибраного біту в контейнер наступним чином: якщо він рівний одиниці – збільшуємо яскравість вибраного пікселя, якщо дорівнює нулю – зменшуємо.

Крок 7. Збільшуємо лічильник фрагментів контейнера на одиницю.

Крок 8. Перевірка умови виходу із циклу, якщо оброблено останній фрагмент контейнеру виходимо з підпрограми, інакше повертаємося до кроку 4.

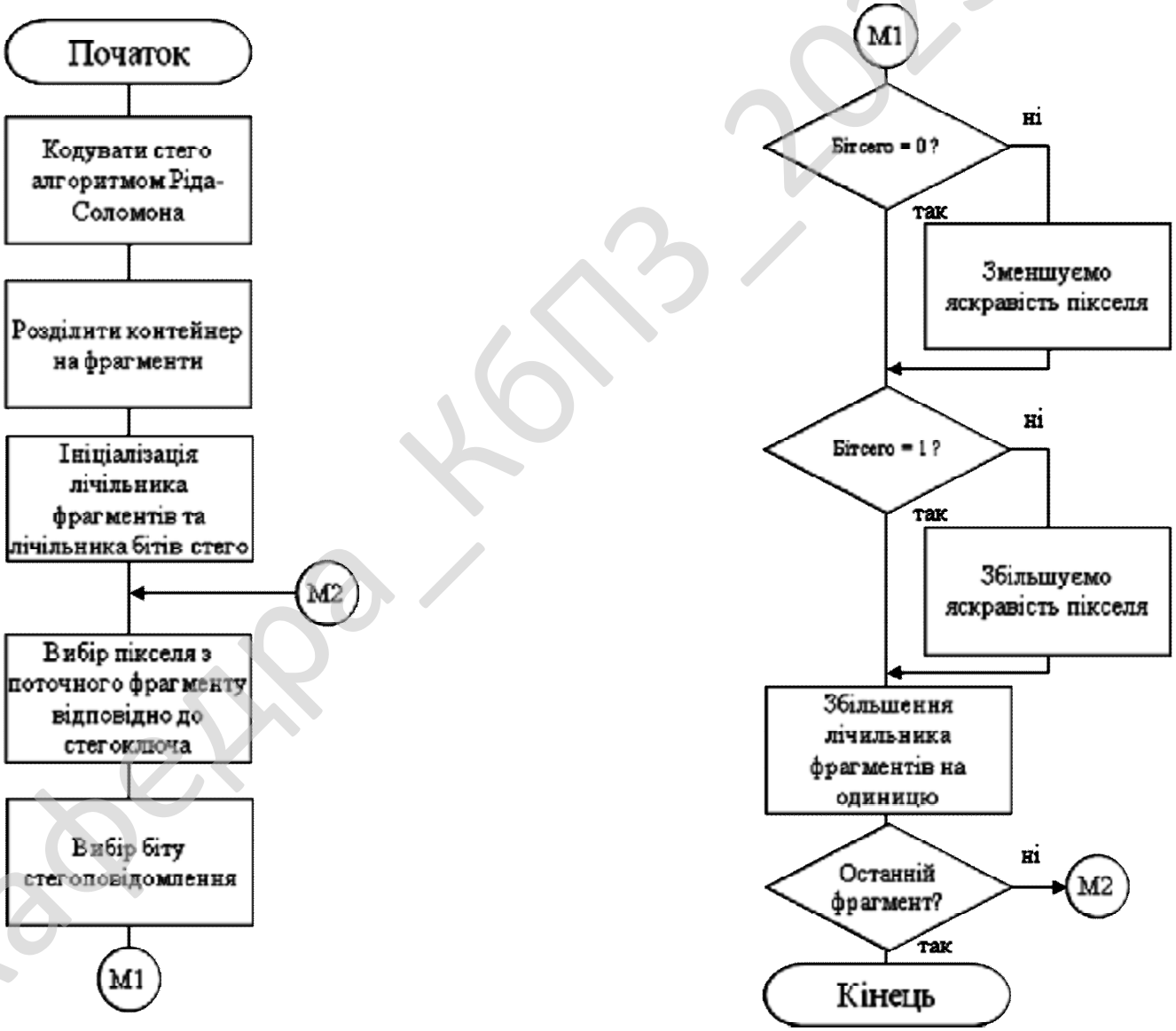

Рисунок 4.2 – Блок-схема підпрограми приховування стегоповідомлення

На рисунку 4.3 зображена блок-схема підпрограми підпрограми читання стегоповідомлення.

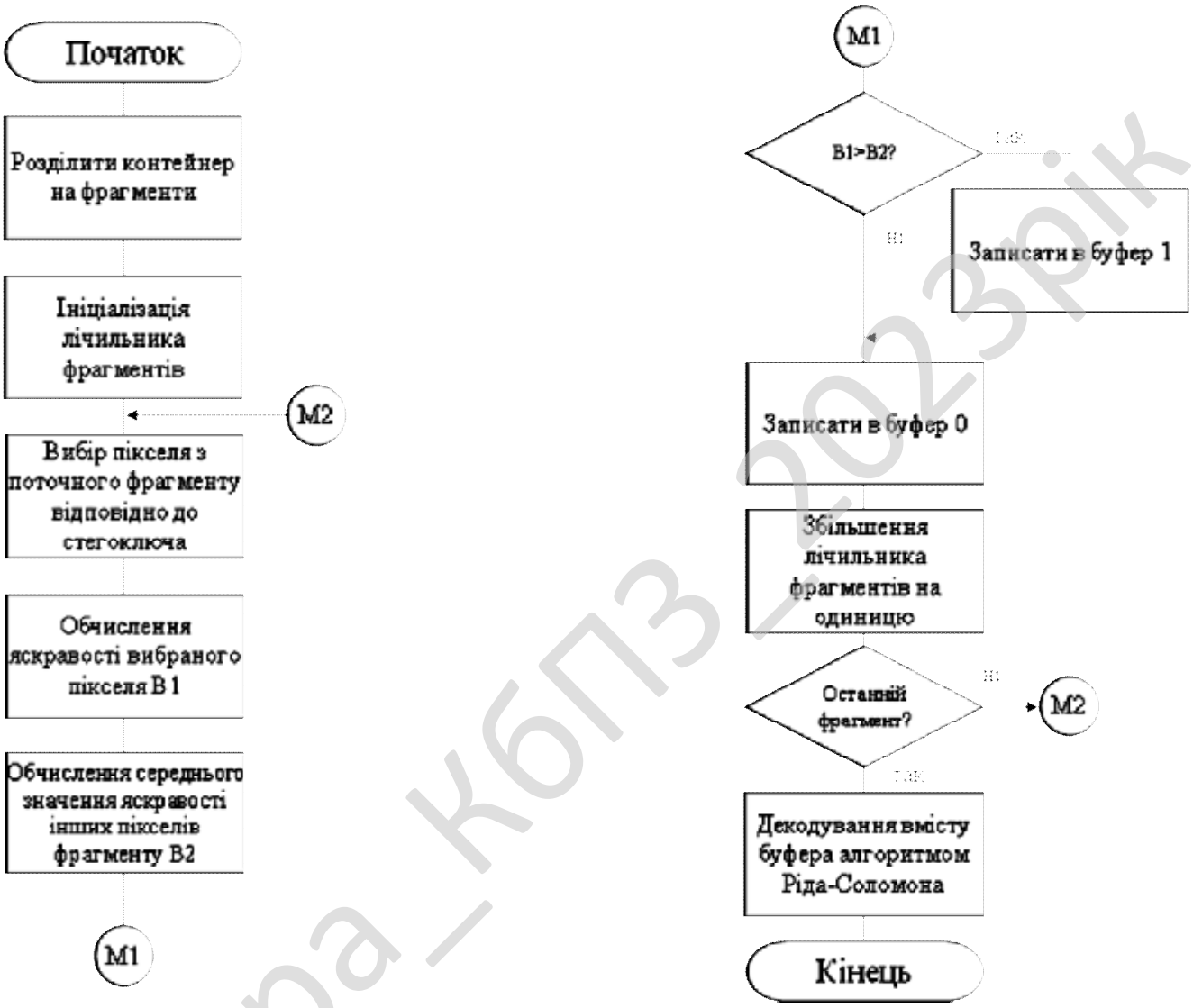

Рисунок 4.3 – Блок-схема підпрограми читання стегоповідомлення

Робота даної підпрограми складається з наступної послідовності кроків:

Крок 1. Розділення контейнеру на фрагменти 8x8.

Крок 2. Ініціалізація лічильника фрагментів.

Крок 3. Вибір пікселя з поточного фрагменту відповідно до стегоключа.

Крок 4. Обчислення яскравості вибраного пікселя B1.

Крок 5. Обчислення середнього значення яскравості інших пікселів

фрагменту В2.

Крок 6. Перевірка чи більше яскравість обраного пікселя, ніж середнє значення яскравості інших пікселів у поточному фрагменті (В1>В2?). Якщо так, то запис в буфер одиниці, інакше запис в буфер нуля.

Крок 7. Збільшення лічильника фрагментів на одиницю.

Крок 8. Перевірка умови виходу із циклу, якщо оброблено останній фрагмент контейнеру виходимо з циклу, інакше повертаємося до кроку 3.

Крок 9. Декодування вмісту буфера алгоритмом Ріда-Соломона. Передача вмісту буфера (вилученого стегоповідомлення) основній програмі.

Наведемо фрагмент коду, що реалізує роботу розробленого стеганографічного алгоритму:

```
procedure seekbmp(poz : integer);   // Встановлення вказівника для читання/запису
стегоданих із зображення
begin
   totpoz:=poz;
   poz:=(poz-1)*8;
   curcol:=(poz mod (form2.bmp.width*3))+1;  // номер байта в графічному рядку
   curline:=poz div (form2.bmp.width*3);    //номер графічного рядка, в якому
знаходиться необхідна позиція
end;
// Dat_ Вказівник на буфер для читання
function  readdata(dat_ : TBuffer_;len : integer): integer;
// LEN – довжина даних
var
   x,y     : integer;
   pt      : TBuffer_;
   dat     : integer;
begin
   pt:=form2.bmp.ScanLine[curline];  // вказівник на потрібний графічний рядок
   for y:=1 to len do
   begin
      for x:=1 to 8 do              // від біта 0 до биту 7 кожного рядка даних
      begin
         if curcol > maxcol then    // перевірка чи не вийшов номер байта в
графічному рядку ща межу
         begin
            inc (curline);
```

```
            pt:=form2.bmp.ScanLine[curline];  // вказівник на наступний рядок
            curcol:=1;
         end;
         // вбудовування інформації
         if ((pt^[curcol] and 1) <> 0) then dat:=dat or (1 shl (x-1)) else
dat:=dat and (not((1 shl (x-1))));
         inc (curcol);
      end;
      dat_^[y]:=byte(dat);
   end;
   decode(dat_,len,totpoz);
   totpoz:=totpoz+len;
end;
function  Writedata(dat_ : TBuffer_;len : integer): integer;
var
   x,y     : integer;
   pt      : TBuffer_;
   d       : integer;
begin
   code(dat_,len,totpoz);  // кодування інформації, що вбудовується
   pt:=form2.bmp.ScanLine[curline];
   for y:=1 to len do
   begin
      d:=dat_^[y];
      for x:=1 to 8 do
      begin
         if curcol > maxcol then
         begin
            inc (curline);
            pt:=form2.bmp.ScanLine[curline];
            curcol:=1;
         end;
         if ((d and 1) <> 0 ) then pt^[curcol]:=(pt^[curcol] or 1) else
pt^[curcol]:=(pt^[curcol] and $FE);
         d:= d shr 1;
         inc (curcol);
      end;
   end;
   totpoz:=totpoz+len;
   writedata:=0;
end;
```

```
function checkbmp: integer;   // перевірка чи є в завантаженому малюнку вбудована
інформація
var
   rt : array [1..4] of byte;
begin
   seekbmp(1);
   readdata(@rt[1],4);
   if (rt[1]=22) and (rt[2]=22) and (rt[3]=77) and (rt[4]=77) then checkbmp:=0
   else checkbmp:=-1;
end;
```

Наведемо фрагмент коду, що здійснює читання стегоповідомлення.

Фрагмент коду, що здійснює шифрування текстового повідомлення:

```
procedure code(dat_ : TBuffer_; len,totpos : integer);
{ Принцип кодування наступний:
   1: Обчислюється контрольна сума пароля
   2: Обчислюється контрольний добуток пароля
   3: Всі закодовані дані представляються як масив байтів
   4: Від кожного байта даних віднімається байт контрольної суми пароля
   5: З результатом попереднього обчислення робиться XOR з байтом контрольного
добутку пароля
   6: До результату попереднього обчислення додається код відповідного символу з
рядка пароля.
      як тільки рядок пароля закінчується знову переходимо на його початок
}
var
   cdcnt,m,d,x,sm : integer;
begin
   if passwrd.password='' then exit; // якщо пароль не введено, то вихід
   sm:=0;
   for x:=1 to length(passwrd.password) do sm:=sm+ord(passwrd.password[x]); //
сума символів пароля
   m:=1;
   for x:=1 to length(passwrd.password) do    // добуток символів пароля
   begin
      m:=m*ord(passwrd.password[x]);
      while ((m and 1) <> 1) do m:= m shr 1;   // Видалення молодших нулів
      while (m > (256*256*128)) do m:= m shr 1; // щоб не було переповнення
   end;
   cdcnt:=totpos mod length(passwrd.password);
   for x:=1 to len do
   begin
```

```
      d:=dat_^[x];
      d:=((d+2048- sm) xor m) + ord (passwrd.password[cdcnt]) ;
      inc (cdcnt);
      if cdcnt>length(passwrd.password) then cdcnt:=length(passwrd.password);
      dat_^[x]:=byte(d);
   end;
end;
```

Фрагмент коду, що здійснює дешифрування текстового повідомлення:

```
procedure decode(dat_ : TBuffer_; len,totpos : integer);
var
   cdcnt,m,d,x,sm : integer;
begin
   if passwrd.password='' then exit;
   sm:=0;
   for x:=1 to length(passwrd.password) do sm:=sm+ord(passwrd.password[x]);
   m:=1;
   for x:=1 to length(passwrd.password) do
   begin
      m:=m*ord(passwrd.password[x]);
      while ((m and 1) <> 1) do m:= m shr 1;
      while (m > (256*256*128)) do m:= m shr 1;
   end;
   cdcnt:=totpos mod length(passwrd.password);
   for x:=1 to len do
   begin
      d:=dat_^[x];
      d:=((d- ord(passwrd.password[cdcnt])) xor m )+ sm;
      inc (cdcnt);
      if cdcnt>length(passwrd.password) then cdcnt:=length(passwrd.password);
      dat_^[x]:=byte(d);
   end;
end;
```

Реалізація алгоритму Ріда-Соломона:

```
procedure tform1.kodhem;
var
     s,bina,hem:string;
     a,b,i,z,b1,b2,b3:integer;
begin
     s:=memo1.Text;
     hem:='';
     z:=length(s);
```

```
      a:=1;
      while a<z do
      begin
            i:=0;
            if s[a]='1' then i:=i xor 3;
            if s[a+1]='1' then i:=i xor 5;
            if s[a+2]='1' then i:=i xor 6;
            if s[a+3]='1' then i:=i xor 7;
            b:=i mod 8;
            b3:=b div 4;
            b:=b mod 4;
            b2:=b div 2;
            b1:=b mod 2;
            bina:=inttostr(b1)+inttostr(b2)+s[a]+inttostr(b3)+s[a+1]+s[a+2]+s[a
            +3];
            hem:=hem+bina;
      a:=a+4;
      end;
      memo2.Text:=hem;
      memo3.Text:=hem;
end;
       //Перетворення у двійковий вигляд
procedure tform1.binar;
var
      a,b,temp,t1:integer;
      bin:string;
begin
      bin:='';
      for a:=1 to dlina do
      begin
            for b:=1 to 2 do
            begin
                  temp:=hex[b,a] div 8;
                  bin:=bin+inttostr(temp);
                  t1:=hex[b,a] mod 8;
                  temp:=t1 div 4;
                  bin:=bin+inttostr(temp);
                  t1:=t1 mod 4;
                  temp:=t1 div 2;
                  bin:=bin+inttostr(temp);
                  temp:=t1 mod 2;
                  bin:=bin+inttostr(temp);
```

```
            end;
      end;
      memo1.Text:=bin;
end;
       //Перетворення у шістнадцятковий вигляд
procedure tform1.kodhex;
var
      s,h,h1,h2:string;
      b,i:integer;
begin
      s:=memo6.Text;
      dlina:=length(s);
      if dlina=0 then exit;
      h:='';
      for b:=1 to dlina do
      begin
            i:=ord(s[b]);
            hex[1,b]:=i div 16;
            h1:=inttostr(hex[1,b]);
            case hex[1,b] of
                  10:h1:='A';
                  11:h1:='B';
                  12:h1:='C';
                  13:h1:='D';
                  14:h1:='E';
                  15:h1:='F';
            end;
            hex[2,b]:=i-(hex[1,b]*16);
            h2:=inttostr(hex[2,b]);
            case hex[2,b] of
                  10:h2:='A';
                  11:h2:='B';
                  12:h2:='C';
                  13:h2:='D';
                  14:h2:='E';
                  15:h2:='F';
            end;
            h:=h+h1+h2+',';
      end;
      delete(h,length(h),1);
      memo5.Text:=h;
end;
```

```
//Підпрограмма визначення кількості помилок та їх виправлення, якщо вони є
procedure TForm1.Button2Click(Sender: TObject);
var
     s:string;
     a,b,i,z,f,osh:integer;
begin
     s:=memo3.Text;
     z:=length(s);
     a:=1;
     osh:=0;
     while a<z do
     begin
           i:=0;
           for f:=0 to 6 do if s[a+f]='1' then i:=i xor (f+1);
           i:=i mod 8;
           if i<>0 then
           begin
                 inc(osh);
                 if s[a+i-1]='0' then s[a+i-1]:='1' else s[a+i-1]:='0';
           end;
           a:=a+7;
           end;
           label8.Caption:='Знайдено помилок    '+inttostr(osh)+' шт.';
           memo4.Text:=s;
     end;
end.
```

4.2 Захист розробленого програмного забезпечення

Захист розробленого програмного забезпечення буде відбуватися за допомогою IDEA – симетричний блоковий алгоритм шифрування даних, запатентований швейцарською фірмою Ascom. Відомий тим, що застосовувався в пакеті програм шифрування PGP. У листопаді 2000 року IDEA був представлений як кандидат у проекті NESSIE в рамках програми Європейської комісії IST (англ. Information Societes Technology, інформаційні громадські технології).

Першу версію алгоритму розробили в 1990 році Лай Сюецзя (Xuejia Lai) і Джеймс Мессі (James Massey) зі Швейцарського інституту ETH Zürich (за

контрактом з Hasler Foundation, яка пізніше влилася в Ascom-Tech AG) як заміна DES (англ. Data Encryption Standard, стандарт шифрування даних) і назвали її PES (англ. Proposed Encryption Standard, запропонований стандарт шифрування). Потім, після публікації робіт Біхамом і Шаміра по диференціальному криптоанализу PES, алгоритм був поліпшений з метою посилення криптостійкості і названий IPES (англ. Improved Proposed Encryption Standard, покращений запропонований стандарт шифрування). Через рік його перейменували в IDEA (англ. International Data Encryption Algorythm).

Так як IDEA використовує 128-бітний ключ і 64-бітний розмір блоку, відкритий текст розбивається на блоки по 64 біт. Якщо таке розбиття неможливо, останній блок доповнюється різними способами певною послідовністю біт. Для уникнення витоку інформації про кожному окремому блоці використовуються різні режими шифрування. Кожен вихідний незашифрований 64 – біт ний блок ділиться на чотири підблока по 16 біт кожен, так як всі алгебраїчні операції, що використовуються в процесі шифрування, відбуваються над 16-бітними числами. Для шифрування і розшифрування IDEA використовує один і той же алгоритм.

Позначення операцій:

- $\boxplus$ Додавання за модулем 2^{16}.
- $\odot$ Множення за модулем $2^{16}+1$.
- $\oplus$ Побітова виключна диз'юнкція.

Фундаментальним нововведенням в алгоритмі є використання операцій з різних алгебраїчних груп, а саме:

Додавання за модулем 2^{16}.

Множення за модулем $2^{16}+1$.

Побітова виключна диз'юнкція (XOR).

Ці три операції несумісні в тому сенсі, що ніякі дві з них не задовольняють дистрибутивному закону, тобто:

$$a \odot (b \oplus c) \neq (a \odot b) \oplus (a \odot c).$$

Застосування цих трьох операцій ускладнює криптоаналіз IDEA в порівнянні з DES, який базується виключно на операції виключає АБО, а також дозволяє відмовитися від використання S-блоків і таблиць заміни. IDEA є модифікацією мережі Фейстеля.

Генерація ключів

З 128-бітного ключа для кожного з восьми раундів шифрування генерується по шість 16-бітних подключів, а для вихідного перетворення генерується чотири 16-бітних підключа. Всього буде потрібно 52 = 8 x 6 + 4 різних подключів по 16 біт кожен. Процес генерації п'ятдесяти двох 16-бітних ключів полягає в наступному:

Насамперед, 128-бітний ключ розбивається на вісім 16-бітних блоків. Це будуть перші вісім подключів по 16 біт кожен – $(K_1^{(1)}K_2^{(1)}K_3^{(1)}K_4^{(1)}K_5^{(1)}K_6^{(1)}K_1^{(2)}K_2^{(2)})$

Потім цей 128-бітний ключ циклічно зсувається вліво на 25 позицій, після чого новий 128-бітний блок знову розбивається на вісім 16-бітних блоків. Це вже наступні вісім подключів по 16 біт кожен – $(K_3^{(2)}K_4^{(2)}K_5^{(2)}K_6^{(2)}K_1^{(3)}K_2^{(3)}K_3^{(3)}K_4^{(3)})$

Процедура циклічного зсуву і розбивки на блоки триває до тих пір, поки не будуть згенеровані всі 52 16-бітних підключа.

Шифрування

Структура алгоритму IDEA показана на рисунку 4.4.

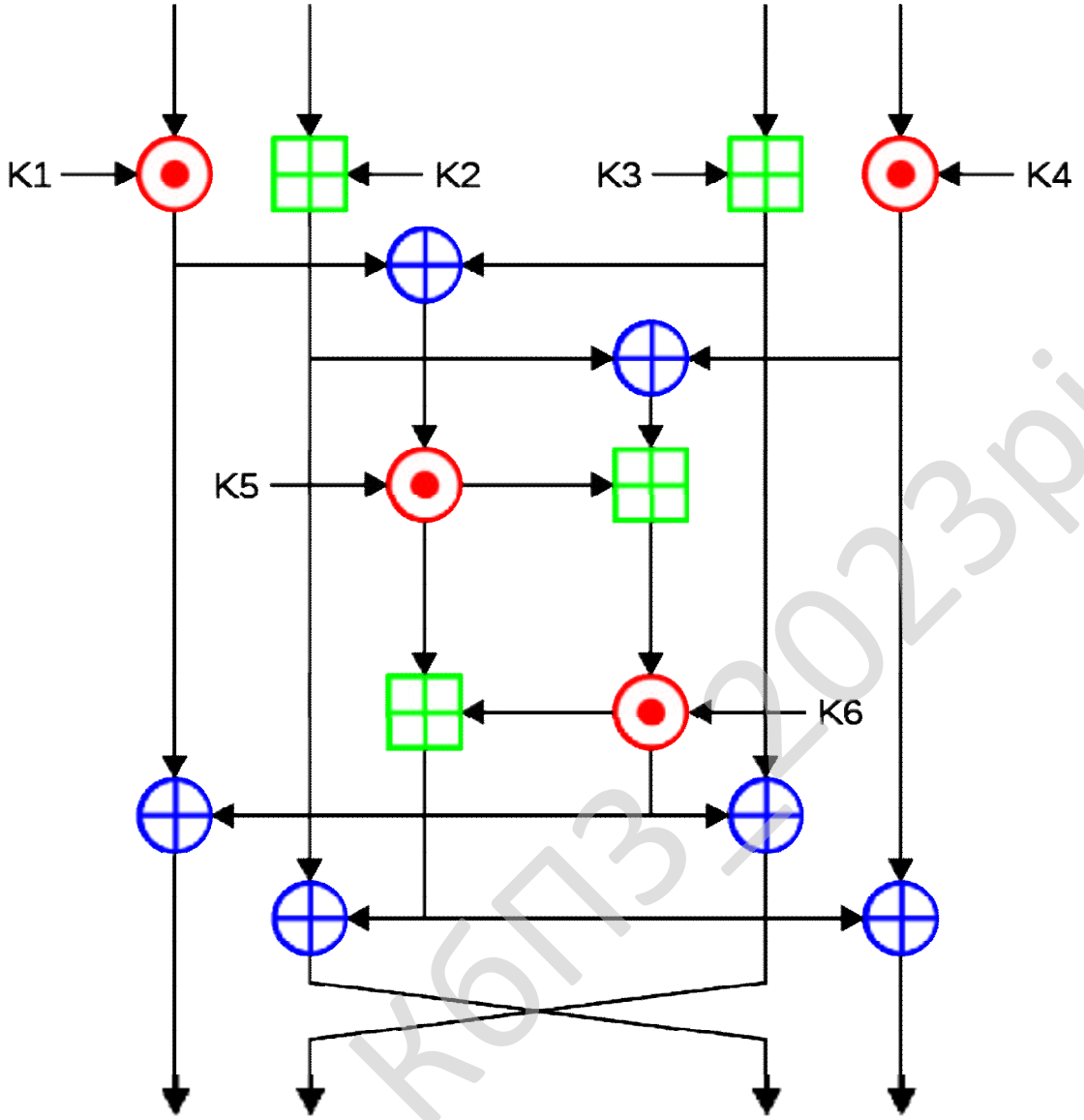

Рисунок 4.4 – Структура алгоритму IDEA

Процес шифрування складається з восьми однакових раундів шифрування і одного вихідного перетворення. Вихідний незашифрований текст ділиться на блоки по 64 біта. Кожен такий блок ділиться на чотири підблока по 16 біт кожен. На рисунку ці підблоки позначені D_1, D_2, D_3, D_4. У кожному раунді використовуються свої підключі згідно з таблицею підключів. Над 16-бітними підключами і підблока незашифрованого тексту проводяться наступні операції:

– Множення за модулем $2^{16}+1 = 65537$, причому замість нуля використовується 2^{16}.

– Додавання за модулем 2^{16}.

– Побітове виключне АБО.

В кінці кожного раунду шифрування є чотири 16-бітних підблоки, які потім використовуються як вхідні підблоки для наступного раунду шифрування. Вихідна перетворення являє собою скорочений раунд, а саме, чотири 16-бітних підблоки на виході восьмого раунду і чотири відповідних підключа піддаються операціям:

– Множення за модулем $2^{16}+1$.

– Додавання за модулем 2^{16}.

Після виконання вихідного перетворення конкатенація підблоків D_1', D_2', D_3' і D_4' являє собою зашифрований текст. Потім береться наступний 64-бітний блок незашифрованого тексту і алгоритм шифрування повторюється. Так продовжується до тих пір, поки не зашифрують всі 64-бітові блоки вихідного тексту.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Розроблена програма має простий та інтуїтивно зрозумілий інтерфейс, який зображений на рисунку 5.1.

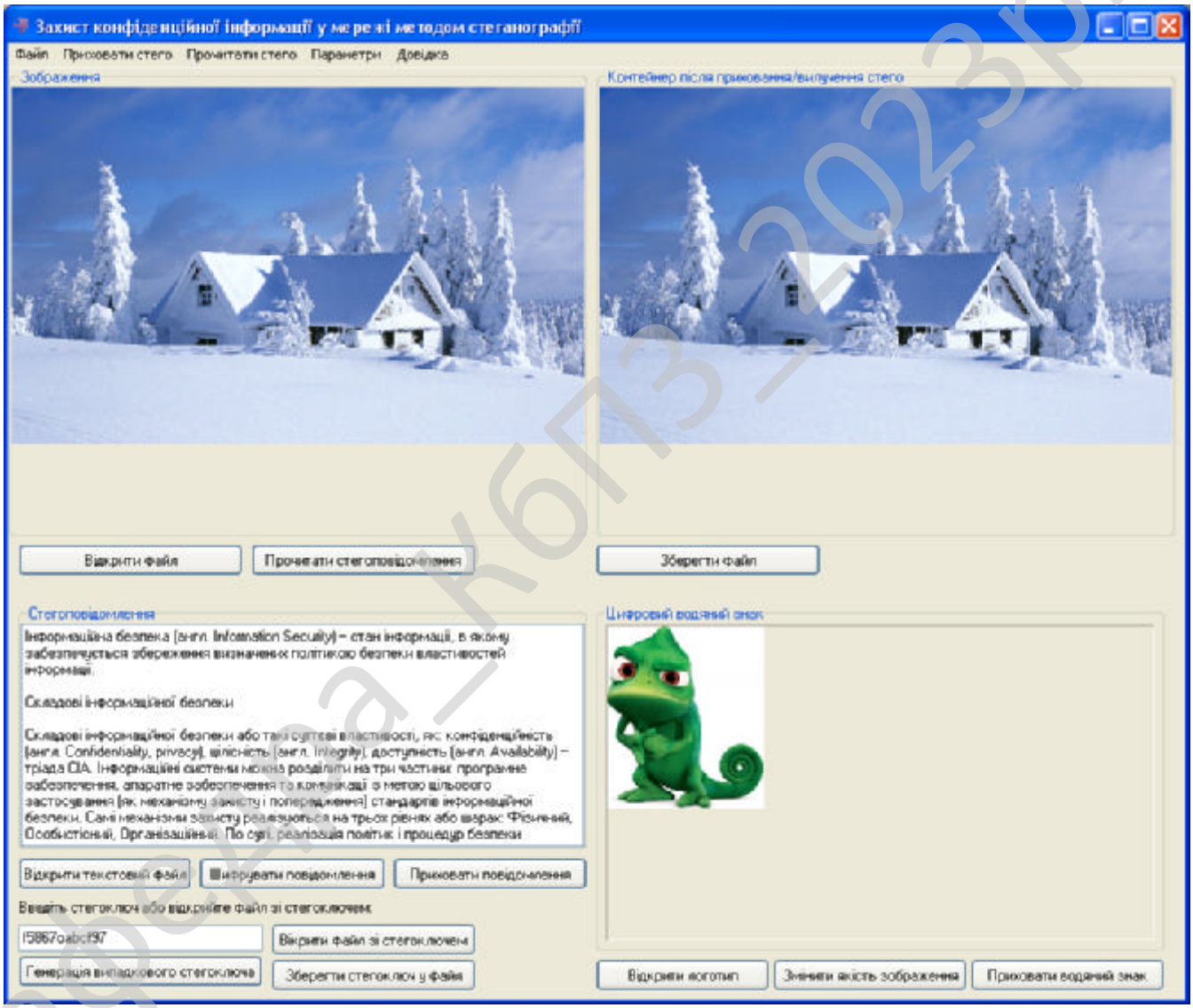

Рисунок 5.1 – Головне вікно програми

Функціональні можливості розробленого програмного забезпечення:

1. Здійснення приховання конфіденційної текстової інформації у цифрове зображення.

2. Здійснення приховання конфіденційної графічної інформації у цифрове зображення.

3. Здійснення вбудовування невидимих цифрових водяних знаків у цифрове зображення з метою захисту авторських прав на нього.

4. Читання з файлу прихованого стегоповідомлення, що може являти собою текстову чи графічну інформацію, або цифрові водяні знаки.

5. Попереднє шифрування текстової інформації перед вбудовуванням.

6. Генерація та збереження у текстовий стегоключів.

Для вбудовування текстового повідомлення у цифрове зображення необхідно:

– відкрити зображення-контейнер;

– ввести вручну або з файлу текстове повідомлення;

– ввести або згенерувати стегоключ;

– за необхідності додаткового захисту, виконати шифрування тексту;

– натиснути кнопку «Приховати повідомлення».

– зберегти заповнене зображення-контейнер у файл, вказавши ім'я і шлях.

Для вбудовування графічного повідомлення (або цифрового водяного знаку) у цифрове зображення необхідно:

– відкрити зображення-контейнер;

– відкрити графічний файл для приховування;

– ввести або згенерувати стегоключ;

– вибрати якість зображення для приховування;

– натиснути кнопку «Приховати водяний знак».

– зберегти заповнене зображення-контейнер у файл, вказавши ім'я і шлях.

Для читання стегоповідомлення з заповненого зображення-контейнеру необхідно:

– відкрити заповнене зображення-контейнер;

– ввести вручну або з файлу стегоключ;

– натиснути кнопку «Прочитати стегоповідомлення», після чого воно буде виведене на екран;

– за необхідності, зберегти очищене від стегоповідомлення зображення-контейнер у файл на диску.

Для більшої наглядності в розробленій програмі зображення-контейнер виводиться на екран в двох екземплярах. Під час вбудовування стегоповідомлення – порожній та заповнений контейнер, під час читання стегоповідомлення – заповнений та очищений контейнер. Це дає змогу побачити, що використаний алгоритм стеганографії дійсно не вносить видимих людиною змін та викривлень у зображення-контейнер.

На рисунку 5.2 наведено вікно довідки, що містить інформацію про розроблене програмне забезпечення.

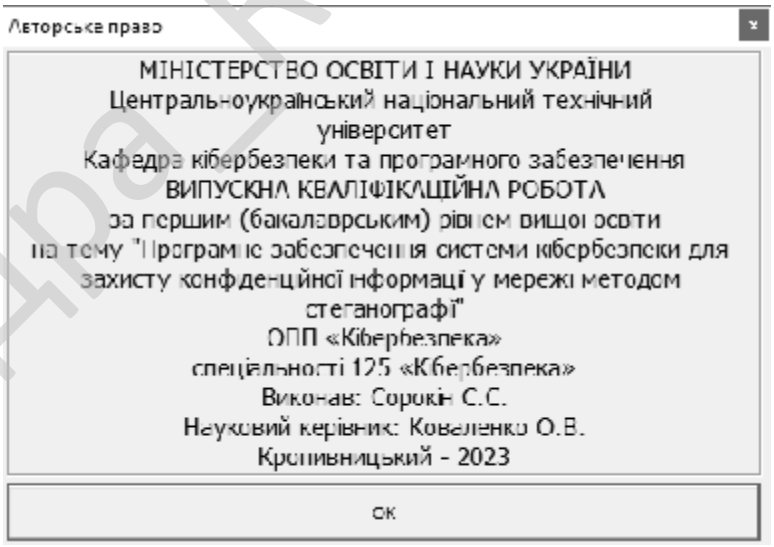

Рисунок 5.2 – Вікно довідки

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем для захисту конфіденційної інформації у мережі методом стеганографії.

– Досліджена система для захисту конфіденційної інформації у мережі методом стеганографії.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання для захисту конфіденційної інформації у мережі методом стеганографії.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об’єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Delphi 10.4.1. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи кібербезпеки для захисту конфіденційної інформації у мережі методом

стеганографії. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи кібербезпеки й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи кібербезпеки Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм IDEA.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Корольов В.Ю.* Планування досліджень методів стеганографії та стеганоаналізу / В.Ю. Корольов, В.В. Поліновський, В.А. Герасименко, М.Л. Горінштейн // Вісник Хмельницького національного університету. Технічні науки. – 2011. – №4. – С. 187-196.

2. *Королев В.Ю.* Стеганография по методу наименее значимого бита на базе персонализированных флеш-накопителей / В.Ю. Королев, В.В. Полиновский, В.А. Герасименко // Управляющие системы и машины. – 2011. – №1(231). – С. 79-87.

3. *Корольов В.Ю.* RS-стеганоаналіз. Принципи роботи, недоліки та концепція метода його обходу / В. Ю. Корольов, В. В. Поліновський, В. А. Герасименко // Вісник Вінницького політехнічного інституту. – 2010. – № 6. – С. 66–71.

4. *Корченко О.Г.* Сучасні квантові технології захисту інформації / О.Г. Корченко, Є.В. Васіліу, С.О. Гнатюк// Науково-технічний журнал «Захист інформації». – 2010, № 1. – С. 77-89

5. *Кошкіна Н.В.* Ефективні спектральні алгоритми для вирішення задач цифрової стеганографії [Текст] : автореф. дис. на здобуття наук. ступеня канд. фіз.-мат. наук : спец. 01.05.01 «Теоретичні основи інформатики та кібернетики» / Н.В. Кошкіна – К. : НАН Укр. Ін-т кібернетики ім. В.М. Глушкова, 2005. – 20 с.

6. *Смирнов А.А.* Ансамблевые свойства недвоичных дискретных сигналов / А.А. Кузнецов, А.А. Смирнов, А.М. Носик, Л.Н. Качур, В.Н. Сай // Системи управління, навігації та зв'язку. – Випуск 4 (8). – К.: ДП «ЦНДІНУ». – 2008. – С. 175-177.

7. *Смирнов А.А.* Разработка метода и алгоритмов синтеза больших ансамблей недвоичных дискретных сигналов на основе обобщенных перестановочных преобразований / А.А. Кузнецов, Ал.М. Носик, А.А. Смирнов,

Л.Н. Качур, Ан.М. Носик // Збірник наукових праць «Системи обробки інформації». – Випуск 5(72). – Х.: ХУПС – 2008. – С. 151-156.

8. *Смирнов А.А.* Формирование дискретных сигналов с многоуровневой функцией корреляции / А.А. Кузнецов, А.А. Смирнов, В.Н. Сай // Збірник наукових праць «Системи обробки інформації». – Випуск 5 (95). – Х.: ХУПС. – 2011. – С. 50-60.

9. *Смирнов А.А.* Дискретные сигналы с многоуровневой функцией корреляции / А.А. Кузнецов, А.А. Смирнов, В.Н. Сай // Радиотехника: Всеукраинский межведомственный научно-технический сборник. Тематический выпуск «Информационная безопасность» – Выпуск 166. – Х.: ХНУРЭ. – 2011. – С. 142-152.

10. *Смирнов А.А.* Математическая модель и структурная схема стеганографической системы / А.А. Кузнецов, А.А. Смирнов, Е.В. Мелешко // Збірник наукових праць Кіровоградського національного технічного університету / техніка в сільськогосподарському виробництві, галузеве машинобудування, автоматизація. Випуск 25. Частина 1. – Кіровоград: КНТУ. – 2012. – С. 273-281.

11. *Смірнов О.А.* Стеганографічне приховування інформації із використанням прямого розширення спектру / О.О. Кузнецов, О.А. Смірнов // Збірник тез доповідей науково-практичної конференції «Застосування інформаційних технологій у підготовці та діяльності сил охорони правопорядку». м. Харків. 21-22 березня 2012 р. – Харків. АВВ МВС. – 2012. – С. 49-52.

12. *Смирнов А.А.* Математическая модель и структурная схема стеганографической системы / А.А. Кузнецов, А.А. Смирнов, Е.В. Мелешко // Збірник тез XIII міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування». м. Кіровоград. 13-14 квітня 2012 р. – Кіровоград: КНТУ. – 2012. – С. 91-92.

13. *Смирнов А.А.* Встраивание данных в контейнеры-изображения с использованием сложных дискретных сигналов / А.А. Кузнецов, А.А. Смирнов // Радиотехника: Всеукраинский межведомственный научно-технический сборник.

Тематический выпуск «Информационная безопасность» – Выпуск 166. – Х.: ХНУРЭ. – 2011. – С. 134-141.

14. *Смірнов О.А.* Дослідження ймовірнісних властивостей стеганографічного захисту інформації із використанням прямого розширення спектру / О.О. Кузнецов, О.А. Смірнов, Л.Т. Пархуць, Ю.М. Рябуха // Системи управління, навігації та зв'язку. – Випуск 1 (21) том 1. – К.: ДП «ЦНДІНУ». – 2012. – С. 115-121.

15. *Смирнов А.А.* Встраивание данных в контейнеры-изображения с использованием сложных дискретных сигналов / А.А. Кузнєцов, А.А. Смирнов // Збірник тез XI міжнародної науково-технічної конференції «Проблеми інформатики и моделювання». м. Харків. 25 листопада 2011 р. – Харків: НАНУ, НТУ «ХПІ», РВНЗ «КГУ». – 2011. – С.42.

16. *Smirnov O.A.* Use of Complex Discrete Signals for Steganographic Information Security / A.A. Kuznetsov, A.A. Smirnov // International Journal of Engineering Practical Education. – Volume 1, Issue 1. – USA, Indiana, Riley: Science and Engineering Publishing Company. – 2012. – P. 21-25.

17. *Кущ А.В.* Использование алгоритмов стеганографии при проведении компьютерно-технической экспертизы / А.В. Кущ // VI Всероссийская межвузовская конференция молодых ученых – СПб: СПбГУ ИТМО, 2009.

18. *Ленков С.В.* Методы и средства защиты информации [Текст]: в 2 т / Ленков С.В., Перегудов Д.А., Хорошко В.А. – К.: Арий, 2008. – Т.2: Информационная безопасность. – 2008. – 344 с.

19. *Лукічов В.В.* Методи та засоби стеганографічного захисту інформації в комп'ютерних системах і мережах на основі вейвлет-перетворень: автореф. дис. канд. техн. наук : 05.13.21 «Системи захисту інформації» / В.В. Лукічов – К.: Нац. авіац. ун-т., 2010. – 20 с.

20. *Мак-Вильямс Ф.Дж.* Теория кодов, исправляющих ошибки / Мак-Вильямс Ф.Дж., Слоэн Н.Дж.А. – М.: Связь, 1979. – 744 с.

21. *Митекин В.А.* Модифицированные методы статистического

стегоанализа бинарных и полутоновых изображений / В.А. Митекин // Компьютерная оптика № 28 – Институт систем обработки изображений РАН: 2005 – 145-151с.

22. *Михайличенко О.В*. Применение стеганографических методов сокрытия информации в неподвижных изображениях / О.В. Михайличенко, А.Г. Коробейников, С.Ю. Каменева // Труды международных научно-технических конференций «Интеллектуальные системы» (IEEE AIS'06) и «Интеллектуальные САПР (CAD-2006) «: в 3 т. М.: Физмалит, 2006. Т.2. – С. 511-515.

23. *Михайличенко О.В.* Повышение устойчивости стеганоалгоритмов частотной области на основе дискретно-косинусного преобразования к внешним воздействиям / О.В. Михайличенко, Н.Н. Прохожев, А.Г Коробейников // Научно-технический вестник СПб ГУ ИТМО – СПб.: СПб ГУ ИТМО, 2009.– вып. 2(60). – С.102 – 104.

24. *Михайличенко О.В.* Алгоритм встраивания цифровых водяных знаков в единичный коэффициент матрицы дискретно-косинусного преобразования / О.В. Михайличенко, Н.Н. Прохожев // Сборник трудов VI Всероссийской конференции молодых ученых. Выпуск 6. Информационные технологии. – СПб. : СПбГУ ИТМО, 2009. – С. 644-648.

25. Надёжность и эффективность в технике. Справочник (в 10 томах). Том 1. Методология. Организация. Терминология / [В.С. Авдуевскнй, И.В. Апполонов, Е.Ю. Барзилович и др.]; под ред. А.И. Рембезы – М.: Машиностроение 1986. – 220 с.

26. Надёжность и эффективность в технике. Справочник (в 10 томах). Том 2. Математические методы в теории надежности и эффективности / [В.В. Белов, Ю.К. Беляев, А.Г. Давтян и др.]; под ред. Б.В.Гнеденко – М.: Машиностроение 1987. – 281 с.

27. Надёжность и эффективность в технике. Справочник (в 10 томах). Том 3. Эффективность технических систем / [В.У. Торбин, Г.Н. Охотников, Е.С. Егоров и др.]; под ред. В.Ф. Уткина и Ю.В. Крючкова – М.: Машиностроение

1988. – 327 с.

28. *Науменко М.І.* Теоретичні основи та методи побудови алгебраїчних блокових кодів: монографія. / М.І. Науменко, Ю.В. Стасєв, О.О. Кузнецов – Х.: ХУ ПС, 2005. – 267 с.

29. *Науменко Н.І.* Теорія сигнально-кодових конструкцій / Н.І. Науменко, Ю.В. Стасев, О.О. Кузнецов, С.П. Євсєєв – Х.:ХУ ПС, 2008р. – 489 С.

30. НД ТЗІ 2.5-004-99 Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу. Затверджено наказом ДСТСЗІ СБ України від 28.04.1999 р., №22.

31. *Нікітенко Л.Л.* Методи побудови стійких стеганосистем [Текст] : автореф. дис. на здоб. наук. ступеня канд. фіз.-мат.наук : [спец.] 01.05.01 «Теор. основи інформатики та кібернетики» / Л.Л. Нікітенко – К. : НАН Укр. Ін-т кібернетики ім. В.М. Глушкова, 2010. – 19 с.

32. *Нікітіна О.Ю.* Комп'ютерні технології побудови систем цифрових водяних знаків [Текст] : автореферат канд. техн. наук, спец.: 01.05.01 – теоретичні основи інформатики та кібернетики / О. Ю. Нікітіна. – К. : НАН Укр. Ін-т кібернетики ім. В.М. Глушкова, 2011. – 19 с.

33. *Носик А.М.* Разработка метода формирования недвоичных псевдослучайных последовательностей для построения дискретных сигналов / А.М. Носик, А.А. Смирнов, Л.Н. Качур // Матеріали першої науково-технічної конференції «МНС України: Сучасний стан та проблемні питання страхового фонду документації, перспективи розвитку та взаємодії». Харків. 25-26 квітня 2008 р. – Харків. НДПКТІМ. – 2008. – С. 46-47.

34. *Паламарчук С.А.* Стеганографічні методи приховування інформації в зображеннях / С.А. Паламарчук, І.В. Бабич // Бизнес и безопасность. – К., 2011. – Вип. 3. – С. 35 – 37.

35. *Паламарчук С.А.* Алгоритм реалізації стеганографічного перетворення інформації в зображеннях / С.А. Паламарчук, І.В. Бабич // VI НПС «Пріоритетні напрямки розвитку ТКС та мереж спеціального призначення» 20- 21.10.2011 р.:

Доповіді та тези доповідей. К.: ВІТІ НТУУ «КПІ», 2011. – С. 160.

36. *Пышкин И.М.* Теория кодового разделения сигналов. – М.: Связь, 1980. – 208 с.

37. *Пометун С.О.* Алгебраїчні атаки на потокові шифратори як узагальнення кореляційних атак / С.О. Пометун // Системні дослідження та інформаційні технології. – 2008. – №2. – С.29-40.

38. *Прокис Дж.* Цифровая связь. Пер. с англ./Под ред. Д. Д. Кловского. М.: Радио и связь, 2000. – 800 с.

39. *Пузыренко А.Ю.* Стеганографическая защита информации с использованием потокового аудио-контейнера / В. П. Бабак, Г. Ф. Конахович, А. Ю. Пузыренко // Безопасность информации в ИТКС-2005 : VIII міжнар. наук.-практ. конф., 11–13 травня 2005 р. : тези доп. – К. : НИЦ ―Тезис, 2005. – С. 11.

40. *Пузыренко А.Ю.* Оценка качества реализации стеганографических алгоритмов / В. П. Бабак, Г. Ф. Конахович, А. Ю. Пузыренко // Безопасность информации в ИТКС-2006: IX міжнар. наук.-практ. конф., 17–19 травня 2006 р. : тези доп. – К. : НИЦ ―Тезис, 2006. – С. 18.

41. *Пузиренко О.Ю.* Представлення і прогнозування ефективності нового протоколу оцінки якості реалізації розроблюваних алгоритмів комп'ютерної стеганографії / О. Ю. Пузиренко, Д. О. Навроцький, Л. П. Дюжаєв // Радіотехніка. Радіоапаратобудування : Зб. наук. пр. – Вип. 34. – К. : НТУУ «КПІ», 2007. – С. 150–156.

42. *Садов В.С.* Обнаружение стеганографического канала передачи данных путем анализа однобитного шума изображения / В.С. Садов, И.Л. Чваркова // Известия Белорусской инженерной академии, № 1 (19)/2, 2005, с. 75-78.

43. *Садов В.С.* Оценка информационных потерь при фильтрации изображений./ В.С. Садов, С.Г. Тихоненко, А.Ф. Чернявский // Информатика. – 2005. – № 3(7). – с. 52-59.

44. *Свердлик М.Б.* Оптимальные дискретные сигналы. – М.: Сов. Радио. – 1975, 200с.

45. *Скляр Б.* Цифровая связь. Теоретические основы и практическое применение. – М.: Вильямс, 2003. – 1104 с.

46. *Сифоров В.И.* Радиоприемные устройства / В.И. Сифоров, И.Н. Амиантов, Ю.Н. Антонов-Антипов, В.П. Васильев, Б.В. Данилов, В.Л. Лебедев, С.С. Судаков, К.А. Щуцкой. – М.:«Советское радио» 1974г.–560с.

47. *Смирнов А.А.* Методы и средства компьютерной стеганографии с применением сложных дискретных сигналов для защиты информации в компьютерных системах и сетях: монография / А.А. Смирнов – К.: Изд. «КОД» – 2012. – 350 с.

48. *Смирнов А.А.* Критерии и показатели эффективности стеганографических систем защиты информации / А.А. Смирнов // Радиотехника: Всеукраинский межведомственный научно-технический сборник. Тематический выпуск «Информационная безопасность» – Выпуск 171. – Х.: ХНУРЭ. – 2012. – С. 189-197.

49. *Smirnov A.A.* Block diagram and formal mathematical definition of steganographic system / A.A. Smirnov // International Journal of Computational Engineering Research (IJCER). – Volume 2, Issue 8. – India. Delhi. – 2012. – P. 90-95.

50. *Смирнов А.А.* Математическая модель и структурная схема стеганографического преобразования информации на основе прямого расширения спектра / А.А. Смирнов // Збірник тез доповідей III міжнародної науково-технічної конференції «Інформаційні технології та захист інформації». м. Харків. 20-21 квітня 2012 р. – Харків: ХНЕУ. – 2012. – С. 211.

51. *Смирнов А.А.* Математическая формализация процедуры стеганографического кодирования и декодирования / А.А. Смирнов // Збірник тез V міжнародної науково-практичної конференції «Інтегровані інтелектуальні робототехнічні комплекси» (ІПРТК-2012). м. Київ. 15-16 травня 2012 р. – К.: НАУ. – 2012. – С. 358-360.

52. *Смірнов О.А.* Аналіз та дослідження стеганографічних систем та протоколів для захисту інформації та інформаційних ресурсів / О.А. Смірнов // Збірник тез III міжнародної науково-практичної конференції «Інформаційні технології та комп'ютерна інженерія». м. Вінниця. 29-31 травня 2012 р. – Вінниця: ВНТУ. – 2012. – С. 164-166.

53. *Смирнов А.А.* Построение стеганографических каналов передачи данных с высокой пропускной способностью в компьютерных сетях / А.А. Смирнов // Збірник тез V міжнародної науково-практичної конференції «Комп'ютерні системи та мережні технології» (CSNT-2012). м. Київ. 13-15 червня 2012 р. – Київ: НАУ. – 2012. – С. 121.

54. *Смірнов О.А.* Дослідження стеганографічного перетворення інформаційних повідомлень для організації скритних каналів передачі даних / О.А. Смірнов // Збірник наукових праць «Системи обробки інформації». – Випуск 2(100). – Х.: ХУПС – 2012. – С. 219-222.

55. *Смірнов О.А.* Дослідження методів стеганографічного перетворення інформації / О.А. Смірнов // Матеріали XII Всеукраїнської наукової інтернет-конференції «Наукові дослідження: зв'язок теорії і практики». м. Тернопіль. 29-30 квітня 2012 р. – Тернопіль: ТНЕУ. – 2012. – С. 35-36.

56. *Смірнов О.А.* Дослідження стеганографічного перетворення інформаційних повідомлень для організації скритних каналів передачі даних / О.А. Смірнов // Збірник тез всеукраїнської науково-практичної конференції «Проблеми інформатики та комп'ютерної техніки» (ПІКТ-2012). м. Чернівці. 3-5 травня 2012 р. – Чернівці: ЧНУ. – 2012. – С. 119-120.

57. *Смірнов О.А.* Теоретичні основи вбудовування інформації в нерухливі зображення з використанням складних дискретних сигналів / О.А. Смірнов // Збірник тез IV міжнародної науково-практичної конференції «Проблеми і перспективи розвитку IT-індустрії». м. Харків. 15-16 листопада 2012 р. – Харків: ХНЕУ. – 2012. – С. 230-231.

58. *Смирнов А.А.* Стеганографическое встраивание данных в неподвижные

изображения методом прямого расширения спектра / А.А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 2(6). – Х.: ХУПС. – 2011. – С.126-129.

59. *Смирнов А.А.* Метод стеганографического встраивания информации в неподвижные изображения с использованием сложных дискретных сигналов и прямого расширения спектра / А.А. Смирнов // Науково-технічний журнал «Захист інформації». – Випуск 4 (53). – К.: НАУ. – 2011 – С.64-70.

60. *Смірнов О.А.* Стеганографічний захист інформації із використанням прямого розширення спектру / О.А. Смірнов // Збірник наукових праць Харківського університету Повітряних Сил. – Випуск 1 (30). – Х.: ХУПС. – 2012. – С. 108-112.

61. *Смірнов О.А.* Метод стеганографічного приховування та вилучення даних в просторовій області зображень із використанням прямого розширення спектру / О.А. Смірнов // Збірник наукових праць «Системи обробки інформації». – Випуск 3(101) том 1. – Х.: ХУПС – 2012. – С. 56-61.

62. *Смирнов А.А.* Стеганографическое встраивание данных в неподвижные изображения методом прямого расширения спектра / А.А. Смирнов // Збірник тез II міжнародної науково-технічної конференції «Сучасні напрями розвитку інформаційно-коммунікаційних технологій та засобів управління». м. Київ-м. Харків. 15-16 грудня 2011 р. – Київ: ДП «ЦНДІ НіУ», Харків: ДП «ХНДІ ТМ», Київ: КДАВТ. – 2011. – С.70-71.

63. *Смирнов А.А.* Метод стеганографического встраивания информации в неподвижные изображения с использованием сложных дискретных сигналов и прямого расширения спектра / А.А. Смирнов // Матеріали III міжнародної науково-практичної конференції «Системний аналіз. Інформатика. Управління (САІУ-2012)». м. Запоріжжя. 14-16 березня 2012 р. – Запоріжжя: КПУ. – 2012. С. 264-266.

64. *Смірнов О.А.* Стеганографічне приховування даних в аудіо-контейнерах із використанням прямого розширення спектру / О.А. Смірнов // Збірник тез XIX науково-практичної конференції «Проблеми створення, розвитку та застосування інформаційних систем спеціального призначення». м. Житомир. 19 квітня 2012 р. – Житомир: ЖВІ НАУ. – 2012. – С. 147-148.

65. *Смирнов А.А.* Встраивание данных в частотную область неподвижных изображений с использованием технологии прямого расширения спектра / А.А. Смирнов // Збірник тез доповідей VIII наукової конференції «Новітні технології – для захисту повітряного простору». м. Харків. 18-19 квітня 2012 р. – Харків. ХУПС. – 2012. – С. 174-175.

66. *Смірнов О.А.* Стеганографічне приховування повідомлень в просторовій області зображень із використанням прямого розширення спектру / О.А. Смірнов // Збірник тез II науково-технічної конференції «Безпека інформаційних технологій» «Information Technology Security» (ITSEC-2012). м. Київ. 24-25 квітня 2012 р. – Київ: НАУ. – 2012. С. 22.

67. *Смірнов О.А.* Дослідження ймовірностних характеристик стеганографічного захисту інформації із використанням прямого розширення спектру / О.А. Смірнов // Збірник наукових праць науково-технічної конференції з міжнародною участю «Комп'ютерне моделювання у наукоємних технологіях» (КМНТ-2012). м. Харків. 24-27 квітня 2012 р. – Харків: ХНУ. – 2012. – С. 400-401.

68. *Смирнов А.А.* Методы широкополосной связи в стеганографии / А.А. Смирнов // Збірник тез V міжнародної науково-практичної конференції «Інформаційна та економічна безпека (INFECO-2012)». м. Харків. 24-26 квітня 2012 р. – Харків: ХНЕУ. – 2012. – С. 135-137.

69. *Смірнов О.А.* Технологія прямого розширення спектру в стеганографії / О.А. Смірнов // Збірник тез першої міжнародної науково-технічної конференції «Захист інформації і безпека інформаційних систем». м. Львів. 31 травня – 01 червня2012 р. – Львів: НУ «ЛП». – 2012. С. 122-123.

70. *Смирнов А.А.* Исследование известных методов синтеза дискретных

сигналов / А.А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 3(9). – Х.: ХУПС. – 2012. – С. 123-126.

71. *Смирнов А.А.* Исследование методов синтеза дискретных сигналов с особыми корреляционными свойствами / А.А. Смирнов // Збірник наукових праць «Системи обробки інформації». – Випуск 9(107). – Х.: ХУПС – 2012. – С. 81-85.

72. *Смирнов А.А.* Сравнительные исследования методов синтеза дискретных сигналов с особыми корреляционными свойствами / А.А. Смирнов, Е.В. Мелешко // Збірник тез V міжнародного науково-технічного симпозіуму «Новітні технології в телекомунікаціях» (ДУІКТ-Карпати-2012) м. Київ. 17-21 січня 2012 р. – Київ: ДУІКТ. – 2012. – С. 80-81.

73. *Smirnov A.A.* Analysis and comparative study of synthesis methods of digital signals with special correlation properties / A.A. Smirnov // International Journal on Communications (IJC). – Volume 1, Issue 1. – USA, Indiana, Riley: Science and Engineering Publishing Company. – 2012. – P. 12-20.

74. *Смирнов А.А.* Формирование больших ансамблей дискретных сигналов на основе методов алгебраического кодирования / А.А. Смирнов, Л.Н. Качур, А.Н. Коваленко // Збірник наукових статей «Управління розвитком» за результатами міжнародної науково-практичної конференції «Стратегії ІТ-технологій в освіті економіці та екології». Випуск 7. – Харків: ХНЕУ. – 2007. – С. 70-71.

75. *Смирнов А.А.* Формирование больших ансамблей дискретных сигналов с использованием избыточных кодов / А.А. Смирнов, Л.Н. Качур // Збірка тез доповідей першої всеукраїнської науково-практичної конференції «Перспективи розвитку озброєння і військової техніки в Збройних силах України». Львів. 04-05 березня 2008 р. Львів. ЛІСВ. – 2008. – С. 219-220.

76. *Смирнов А.А.* Синтез больших ансамблей дискретных сигналов с использованием недвоичных избыточных кодов / А.А. Смирнов, А.М. Носик, Л.Н. Качур, С.Ю. Стасев // Матеріали IV наукової конференції Харківського Університету Повітряних Сил імені Івана Кожедуба. Харків.

16-17 квітня 2008 р.– Харків. ХУПС. – 2008. – С. 154.

77. *Смирнов А.А.* Дискретні сигнали з багаторівневою функцією кореляції / А.А. Смирнов // Збірник тез XII міжнародного науково-практичного семінару «Комбінаторні конфігурації та їх застосування». м. Кіровоград. 14-15 жовтня 2011 р. – Кіровоград: КНТУ. – 2011. – С. 119-120.

78. *Смирнов А.А.* Формирование дискретных сигналов с многоуровневой функцией корреляции / А.А. Смирнов, А.А. Кузнецов, В.Н. Сай // Збірник тез IX міжнародної науково-практичної конференції «Математичне та програмне забезпечення інтелектуальних систем» м. Дніпропетровськ. 23-25 листопада 2011р. – Дніпропетровськ: ДНУ. – 2011. – С.247-248.

79. *Смірнов О.А.* Синтез ансамблів дискретних сигналів для стеганографічних систем з прямим розширенням спектру / О.А. Смірнов // Збірник тез XX міжнародної науково-практичної конференції «Інформаційні технології: наука, техніка, технологія, освіта, здоров'я» (MicroCAD-2012). м. Харків. 15-17 травня 2012 р. – Харків: НТУ «ХПІ». – 2012. – С. 45-46.

80. *Смірнов О.А.* Дослідження ансамблевих, кореляційних і структурних властивостей складних дискретних сигналів, які використовуються для побудови стеганографічних систем / О.А. Смірнов, Є.В. Мелешко // Збірник тез IV науково-технічної конференції «Захист інформації з обмеженим доступом та автоматизація її обробки» (PIRAT-2012). м. Київ. 9-10 лютого 2012 р. – Київ: НАУ. – 2012. – С. 32-33.

81. *Смирнов А.А.* Предложения по реализации устройств формирования дискретных сигналов с многоуровневой функцией корреляции / А.А. Смирнов // Науково-технічний журнал «Захист інформації». – Випуск 4 (57). – К.: НАУ. – 2012 – С.94-105.

82. *Smirnov A.A.* The hardware implementation of devices forming discrete signals with multi-level correlation function / A.A. Smirnov // International Journal of Information and Computer Science (IJICS). – Volume 2, Issue 1. – USA, Indiana, Riley: Science and Engineering Publishing Company. – 2013. – P. 1-7.

83. *Смирнов А.А.* Обоснование предложений по реализации динамического режима функционирования радиосистем управления с множественным доступом / А.А. Смирнов, В.Н. Сай, А.В. Коваленко // Системи управління, навігації та зв'язку. – Випуск 3(23). – К.: ДП «ЦНДІНУ». – 2012. – С. 255-262.

84. *Смирнов А.А.* Анализ перспективных направлений в совершенствовании радиосистем управления и связи с организацией множественного доступа / А.А. Смирнов, В.Н. Сай, А.В. Коваленко // Системи озброєння і військова техніка. – Випуск 3(31) – Х.: ХУПС – 2012. – С. 218-226.

85. *Смирнов А.А.* Обоснование критериев и показателей выбора ансамблей дискретных сигналов для радиосистем управления с множественным доступом / А.А. Смирнов // Системи озброєння і військова техніка. – Випуск 4(32) – Х.: ХУПС – 2012. – С. 158-161.

86. *Смирнов А.А.* Исследование абонентской емкости и помехоустойчивости радиоканалов управления с использованием дискретных сигналов с многоуровневой функцией корреляции / А.А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 1(10). – Х.: ХУПС . – 2013. – С. 111-115.

87. *Смірнов О.А.* Дослідження методів стегоаналізу цифрових зображень / О.А. Смірнов, Є.В. Мелешко // Збірник тез науково-практичної конференції «Захист інформації в інформаційно-комунікаційних системах». м. Київ. 24-27 квітня 2012 р. – Київ: НАУ. – 2012. – С. 75-77.

88. *Смірнов О.А.* Дослідження методів стегоаналізу цифрових зображень / О.А. Смірнов, Є.В. Мелешко // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 2(8). – Х.: ХУПС. – 2012. – С. 92-99.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за першим (бакалаврським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 13-02 від 5.01.2023 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи кібербезпеки з ОС та з користувачем;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп'ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14" 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Delphi 10.4.1.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Пояснювальна записка – 88 аркушів.

8 Етапи розробки

8.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти.

8.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

9 Порядок контролю та приймання

9.1 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на попередній захист 23.05.2023 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 8.06.2023 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

першим (бакалаврським) рівнем вищої освіти

_____ Коваленко О.В.

Програмне забезпечення системи кібербезпеки для захисту конфіденційної інформації у мережі методом стеганографії

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 28

Літера: РП

Кропивницький – 2023 року

ABOUT.PAS - довідка

```
unit about;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, jpeg, ExtCtrls;

type
  TFmAbout = class(TForm)
    Memo1: TMemo;
    Button1: TButton;
    Image1: TImage;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FmAbout: TFmAbout;

implementation

{$R *.dfm}

procedure TFmAbout.FormCreate(Sender: TObject);
begin
Memo1.Clear;
Memo1.Lines.Add('БАКАЛАВРСЬКА РОБОТА');
Memo1.Lines.Add('');
Memo1.Lines.Add('на  тему:');
Memo1.Lines.Add('');
Memo1.Lines.Add('Програмне забезпечення системи кібербезпеки для захисту
конфіденційної інформації у мережі методом стеганографії');
Memo1.Lines.Add('');
Memo1.Lines.Add('Керівник:  Коваленко О.В. ');
Memo1.Lines.Add('');
Memo1.Lines.Add('Розробив:  студент   Сорокін Сергій Сергійович');
Memo1.Lines.Add('           гр. КБ-20-3СК ');
Memo1.Lines.Add('');
Memo1.Lines.Add('м. Кропивницький 2023');
Memo1.Lines.Add('');
end;

procedure TFmAbout.Button1Click(Sender: TObject);
begin
FmAbout.Close;
end;
end.
```

STEGOGRAPHY_FOR_NET.PAS – реалізація алгоритму стеганографії

```
unit STEGOGRAPHY_FOR_NET;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, password__STEGOGRAPHY_FOR_NET, Menus, ExtCtrls;

type
  buffer_ = array [1..1024*1024*2] of byte;
  TBuffer_ = ^buffer_;
  fhandle  = record
               name  : shortstring;
               size  : integer;
               next  : byte;
               data_ : TBuffer_;
             end;

  TForm2 = class(TForm)
    ListBox1: TListBox;
    Label1: TLabel;
    ListBox2: TListBox;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Button1: TButton;
    Button2: TButton;
    Label5: TLabel;
    Label6: TLabel;
    PopupMenu1: TPopupMenu;
    N1: TMenuItem;
    N2: TMenuItem;
    N3: TMenuItem;
    N4: TMenuItem;
    ldfl: TOpenDialog;
    svfl: TSaveDialog;
    extr: TOpenDialog;
    Panel1: TPanel;
    N5: TMenuItem;
    N6: TMenuItem;
    procedure loaddir;
    procedure FormCreate(Sender: TObject);
    procedure PopupMenu1Popup(Sender: TObject);
    procedure N2Click(Sender: TObject);
    procedure N4Click(Sender: TObject);
    procedure N1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure N3Click(Sender: TObject);
    procedure Panel1DblClick(Sender: TObject);
    procedure N5Click(Sender: TObject);
    procedure N6Click(Sender: TObject);
  private
    { Private declarations }
  public
     bmp : TBitMap;
    { Public declarations }
  end;

var
  Form2: TForm2;
  estlen,    // Залишилось вільного місця
  filelen,
  datalen,
  maxlen,
  maxcol : integer;
```

```
function  readdata(dat_ : TBuffer_;len : integer): integer;
function  Writedata(dat_ : TBuffer_;len : integer): integer;
function checkbmp: integer;

implementation

uses RS_CODE;

{$R *.DFM}
var
   fcnt,ft2cnt,present,
   totpoz,    // Номер наступного байту при читанні чи записі (для процедур
ReadData та WriteData)
   curline,   // Поточний графічний рядок
   curcol : integer;  // Номер байта в графічному рядку куди буде
записано/прочитано наступний байт
   FAT_ : array [1..100] of Fhandle;
   ft2  : array [1..100] of Fhandle;  // Масив всього, що потрібно занести у
зображення
// Процедура кодування даних
procedure code(dat_ : TBuffer_; len,totpos : integer);
{ Принцип кодування наступний:
   1: Обчислюється контрольна сума пароля
   2: Обчислюється контрольний добуток пароля
   3: Всі закодовані дані представляються як масив байтів
   4: Від кожного байта даних віднімається байт контрольної суми пароля
   5: З результатом попереднього обчислення робиться XOR з байтом контрольного
добутку пароля
   6: До результату попереднього обчислення додається код відповідного символу
з рядка пароля.
      як тільки рядок пароля закінчується знову переходимо на його початок
}
var
   cdcnt,m,d,x,sm : integer;
begin
   if passwrd.password='' then exit; // якщо пароль не введено, то вихід
   sm:=0;
   for x:=1 to length(passwrd.password) do sm:=sm+ord(passwrd.password[x]); //
сума символів пароля
   m:=1;
   for x:=1 to length(passwrd.password) do    // добуток символів пароля
   begin
      m:=m*ord(passwrd.password[x]);
      while ((m and 1) <> 1) do m:= m shr 1;   // Видалення молодших нулів
      while (m > (256*256*128)) do m:= m shr 1; // щоб не було переповнення
   end;

   cdcnt:=totpos mod length(passwrd.password);
   for x:=1 to len do
   begin
      d:=dat_^[x];
      d:=((d+2048- sm) xor m) + ord (passwrd.password[cdcnt]) ;
      inc (cdcnt);
      if cdcnt>length(passwrd.password) then cdcnt:=length(passwrd.password);
      dat_^[x]:=byte(d);
   end;
end;

// розкодування даних
procedure decode(dat_ : TBuffer_; len,totpos : integer);
var
   cdcnt,m,d,x,sm : integer;
begin
   if passwrd.password='' then exit;
   sm:=0;
   for x:=1 to length(passwrd.password) do sm:=sm+ord(passwrd.password[x]);
   m:=1;
   for x:=1 to length(passwrd.password) do
```

```
   begin
      m:=m*ord(passwrd.password[x]);
      while ((m and 1) <> 1) do m:= m shr 1;
      while (m > (256*256*128)) do m:= m shr 1;
   end;

   cdcnt:=totpos mod length(passwrd.password);
   for x:=1 to len do
   begin
      d:=dat_^[x];
      d:=((d- ord(passwrd.password[cdcnt])) xor m )+ sm;
      inc (cdcnt);
      if cdcnt>length(passwrd.password) then cdcnt:=length(passwrd.password);
      dat_^[x]:=byte(d);
   end;
end;

procedure seekbmp(poz : integer);   // Встановлення вказівника для
читання/запису стегоданих із зображення

begin
   totpoz:=poz;
   poz:=(poz-1)*8;
   curcol:=(poz mod (form2.bmp.width*3))+1;  // номер байта в графічному рядку
   curline:=poz div (form2.bmp.width*3);     //номер графічного рядка, в якому
знаходиться необхідна позиція
end;

// Dat_ Вказівник на буфер для читання
function  readdata(dat_ : TBuffer_;len : integer): integer;
// LEN - довжина даних
var
   x,y     : integer;
   pt      : TBuffer_;
   dat     : integer;
begin
   pt:=form2.bmp.ScanLine[curline];  // вказівник на потрібний графічний рядок
   for y:=1 to len do
   begin
      for x:=1 to 8 do               // від біта 0 до биту 7 кожного рядка
даних
      begin
         if curcol > maxcol then    // перевірка чи не вийшов номер байта в
графічному рядку ща межу
         begin
            inc (curline);
            pt:=form2.bmp.ScanLine[curline];  // вказівник на наступний рядок
            curcol:=1;
         end;
         // вбудовування інформації
         if ((pt^[curcol] and 1) <> 0) then dat:=dat or (1 shl (x-1)) else
dat:=dat and (not((1 shl (x-1))));
         inc (curcol);
      end;
      dat_^[y]:=byte(dat);
   end;
   decode(dat_,len,totpoz);
   totpoz:=totpoz+len;
end;

function  Writedata(dat_ : TBuffer_;len : integer): integer;
var
   x,y     : integer;
   pt      : TBuffer_;
   d       : integer;
begin
   code(dat_,len,totpoz);  // кодування інформації, що вбудовується
   pt:=form2.bmp.ScanLine[curline];
   for y:=1 to len do
```

```
   begin
      d:=dat_^[y];
      for x:=1 to 8 do
      begin
         if curcol > maxcol then
         begin
            inc (curline);
            pt:=form2.bmp.ScanLine[curline];
            curcol:=1;
         end;
         if ((d and 1) <> 0 ) then pt^[curcol]:=(pt^[curcol] or 1) else
pt^[curcol]:=(pt^[curcol] and $FE);
         d:= d shr 1;
         inc (curcol);
      end;
   end;
   totpoz:=totpoz+len;
   writedata:=0;
end;

function checkbmp: integer;   // перевірка чи є в завантаженому малюнку
вбудована інформація
var
   rt : array [1..4] of byte;
begin
   seekbmp(1);
   readdata(@rt[1],4);
   if (rt[1]=22) and (rt[2]=22) and (rt[3]=77) and (rt[4]=77) then checkbmp:=0
   else checkbmp:=-1;
end;

procedure TForm2.loaddir;
var
   x : integer;
   hd : Fhandle;
   s : string;
begin
   listbox1.items.clear;
   listbox2.items.clear;
   for x:=1 to present do freemem(fat_[x].data_,fat_[x].size);
   present:=0;
   ft2cnt:=0;
   estlen:=maxlen;
   if checkbmp=0 then
   begin
      seekbmp(5);
      hd.next:=1;
      while hd.next<>0 do
      begin
         inc(present);
         fat_[present].name:='                ';
         readdata(@fat_[present].name[1],16);
         readdata(@fat_[present].size,4);
         readdata(@hd.next,1);
         getmem(fat_[present].data_,fat_[present].size);
         readdata(fat_[present].data_,fat_[present].size);
         s:=inttostr(fat_[present].size);
         while length(s)<7 do s:=' '+s;
         listbox2.items.add(fat_[present].name+'          '+s+'
'+inttostr(present));
         estlen:=estlen-fat_[present].size-21;
      end;
   end;
   label3.caption:=inttostr(estlen);
end;

procedure TForm2.FormCreate(Sender: TObject);
begin
   present:=0;
```

```
end;

function chsel (list : tlistbox):integer;
var
   ok,x : integer;
begin
   ok:=0;
   for x:=1 to list.items.count do if list.selected[x-1] then ok:=x;
   chsel:=ok;
end;

procedure TForm2.PopupMenu1Popup(Sender: TObject);
begin
   popupmenu1.items[0].enabled:=(popupmenu1.PopupComponent.name='ListBox2');
   popupmenu1.items[1].enabled:=(popupmenu1.PopupComponent.name='ListBox2');
   popupmenu1.items[3].enabled:=(popupmenu1.PopupComponent.name='ListBox1');
   if panel1.color <> clblack then popupmenu1.items[0].enabled:=false;
end;

procedure TForm2.N2Click(Sender: TObject);
var
   x,y : integer;
   s : string;
begin
   if popupmenu1.PopupComponent.name='ListBox2' then
   begin
      if chsel(listbox2)=0 then exit;
      if length(listbox2.items[listbox2.itemindex])>36 then
      begin

estlen:=estlen+21+FAT_[strtoint(copy(listbox2.items[listbox2.itemindex],length
(listbox2.items[listbox2.itemindex])-2,3))].size;
         listbox1.items.add(listbox2.items[listbox2.itemindex]);
      end
      else
      begin
        for x:= 1 to ft2cnt do
        begin
           s:=ft2[x].name;
           while pos('\',s) <> 0 do delete (s,1,pos('\',s));
           if length(s)>16 then setlength(s,16);
           while length(s)<16 do s:=s+' ';
           if s=copy(listbox2.items[listbox2.itemindex],1,16) then
           begin
              estlen:=estlen+21+ft2[x].size;
              for y:=x to ft2cnt-1 do ft2[y]:=ft2[y+1];
              dec(ft2cnt);
              break;
           end;
        end;
      end;
      listbox2.items.delete(listbox2.itemindex);
      label3.caption:=inttostr(estlen);
   end;
end;

procedure TForm2.N4Click(Sender: TObject);
begin
   if chsel(listbox1)=0 then exit;
   if estlen <
(21+FAT_[strtoint(copy(listbox1.items[listbox1.itemindex],length(listbox1.item
s[listbox1.itemindex])-2,3))].size) then
   begin
      application.messagebox('Відновлення неможливе так як в малюнку не
вистачає місця','',$10);
      exit;
   end;
   listbox2.items.add(listbox1.items[listbox1.itemindex]);
```

```
   estlen:=estlen-21-
FAT_[strtoint(copy(listbox1.items[listbox1.itemindex],length(listbox1.items[li
stbox1.itemindex])-2,3))].size;
   listbox1.items.delete(listbox1.itemindex);
   label3.caption:=inttostr(estlen);
end;

procedure TForm2.N1Click(Sender: TObject);
var
   f : file;
   s,s1 : string;
begin
   if not ldfl.execute then exit;
   assignfile(f,ldfl.filename);
   filemode:=0;
   if ioresult <> 0 then ;
   {$I-}
   reset(f,1);
   if ioresult <> 0 then
   begin
      application.messagebox('Неможливо відкрити вказаний файл','',$10);
      exit;
   end;
   inc (ft2cnt);
   if estlen < filesize(f) then
   begin
      application.messagebox('Не вистачає вільного місця','',$10);
      closefile(f);
      exit;
   end;
   ft2[ft2cnt].name:=ldfl.filename;
   ft2[ft2cnt].size:=filesize(f);
   closefile(f);
   s:=ldfl.filename;
   while pos('\',s) <> 0 do delete (s,1,pos('\',s));
   if length(s)>16 then setlength(s,16);
   while length(s)<16 do s:=s+' ';
   s1:=inttostr(ft2[ft2cnt].size);
   while length(s)<7 do s:=' '+s;
   listbox2.items.add(s+'         '+s1);
   estlen:=estlen-ft2[ft2cnt].size-21;
   label3.caption:=inttostr(estlen);
end;

procedure TForm2.Button2Click(Sender: TObject);
begin
   close;
end;

procedure savedata(hdl : FHandle);  // запис інформації в малюнок
var
   hdl1 : FHandle;
begin
   getmem(hdl1.data_,hdl.size);
   move(hdl.data_^,hdl1.data_^,hdl.size);
   hdl1.size:=hdl.size;
   hdl1.name:=hdl.name;
   while pos('\',hdl1.name)<> 0 do delete (hdl1.name,1,pos('\',hdl1.name));
   while length(hdl1.name)<16 do hdl1.name:=hdl1.name+' ';
   if fcnt=form2.listbox2.items.count then hdl.next:=0 else hdl.next:=255;
   hdl1.next:=hdl.next;
   writedata(@hdl1.name[1],16);
   writedata(@hdl1.size,4);
   writedata(@hdl1.next,1);
   writedata(hdl1.data_,hdl.size);
   freemem(hdl1.data_,hdl.size);
   inc (fcnt);
end;
```

```
procedure TForm2.Button1Click(Sender: TObject);
var
    hh : array [1..4] of byte;
    x,y : integer;
    s : shortstring;
    f : file;
begin
   if listbox2.items.count>0 then
   begin
      hh[1]:=24;
      hh[2]:=06;
      hh[3]:=19;
      hh[4]:=77;
   end;
   seekbmp(1);
   writedata(@hh[1],4);
   fcnt:=1;
   for x:=1 to listbox2.items.count do if length(listbox2.items[x-1])>37 then
      savedata(FAT_[strtoint(copy(listbox2.items[x-1],length(listbox2.items[x-
1])-2,3))]);
   for x:=1 to listbox2.items.count do if length(listbox2.items[x-1])<37 then
   begin
      for y:= 1 to ft2cnt do
      begin
         s:=ft2[y].name;
         while pos('\',s) <> 0 do delete (s,1,pos('\',s));
         if length(s)>16 then setlength(s,16);
         while length(s)<16 do s:=s+' ';
         if s=copy(listbox2.items[x-1],1,16) then
         begin
            assignfile(f,ft2[y].name);
            if ioresult <> 0 then ;
            {I-}
            filemode:=0;
            reset(f,1);
            if ioresult <> 0 then
            begin
               s:='Неможливо відкрити файл'+ft2[y].name+#0;
               application.messagebox(@s[1],'',$10);
               exit;
            end;
            getmem(ft2[y].data_,ft2[y].size);
            blockread(f,ft2[y].data_^,ft2[y].size);
            closefile(f);
            savedata(ft2[y]);
            freemem(ft2[y].data_,ft2[y].size);
            break;
         end;
      end;
   end;
   if not svfl.execute then exit;
   bmp.savetofile(svfl.filename);
end;

procedure TForm2.N3Click(Sender: TObject);
var
   s : shortstring;
   f : file;
begin
   if popupmenu1.popupcomponent.name='ListBox2' then
   begin
      if chsel(listbox2)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end
   else
   begin
      if chsel(listbox1)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end;
```

```
   if length(s)<36 then exit;
   extr.filename:=FAT_[strtoint(copy(s,length(s)-2,3))].name;
   if not extr.execute then exit;
   assignfile(f,extr.filename);
   if ioresult <> 0 then;
   filemode:=2;
   {$I-}
   rewrite(f,1);
   if ioresult <> 0 then
   begin
      application.messagebox(Неможливо створити вказаний файл','',$10);
      exit;
   end;
   blockwrite(f,FAT_[strtoint(copy(s,length(s)-
2,3))].data_^,FAT_[strtoint(copy(s,length(s)-2,3))].size);
   closefile(f);
end;

procedure TForm2.Panel1DblClick(Sender: TObject);
begin
   panel1.color:=clblack;
end;

procedure TForm2.N5Click(Sender: TObject);
var
   x,y : integer;
   s : string;
begin
   if popupmenu1.popupcomponent.name='ListBox2' then
   begin
      if chsel(listbox2)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end
   else
   begin
      if chsel(listbox1)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end;

   if length(s)<36 then exit;
   form1.Memo2.lines.clear;
   y:=strtoint(copy(s,length(s)-2,3));
   s:='';
   x:=1;
   while x<= FAT_[y].size do
   begin
      if FAT_[y].data_^[x]<>13 then s:=s+chr(FAT_[y].data_^[x])
      else
      begin
         form1.Memo2.lines.add(s);
         s:='';
         inc(x);
      end;
      inc(x);
   end;
   if s<>'' then form1.Memo2.lines.add(s);
   form1.fmode:=2;
   form1.Button1Click(nil);
   form1.showmodal;
end;

procedure TForm2.N6Click(Sender: TObject);
var
   x,y : integer;
   s : string;
begin
   if popupmenu1.popupcomponent.name='ListBox2' then
   begin
```

```
      if chsel(listbox2)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end
   else
   begin
      if chsel(listbox1)=0 then exit;
      s:=listbox2.items[listbox2.itemindex];
   end;

   if length(s)<36 then exit;
   form1.richedit1.lines.clear;
   y:=strtoint(copy(s,length(s)-2,3));
   s:='';
   x:=1;
   while x<= FAT_[y].size do
   begin
      if FAT_[y].data_^[x]<>13 then s:=s+chr(FAT_[y].data_^[x])
      else
      begin
         form1.richedit1.lines.add(s);
         s:='';
         inc(x);
      end;
      inc(x);
   end;
   if s<>'' then form1.richedit1.lines.add(s);
   form1.fmode:=2;
   form1.showmodal;
end;

end.
```

PASSWORD__STEGOGRAPHY_FOR_NET.PAS - модуль створення стегоключа

```
unit password__STEGOGRAPHY_FOR_NET;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls;

type
  Tpasswrd = class(TForm)
    Edit1: TEdit;
    Label1: TLabel;
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
     password : shortstring;
    { Public declarations }
  end;

var
  passwrd: Tpasswrd;

implementation

{$R *.DFM}

procedure Tpasswrd.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
   if key=#13 then
   begin
      password:=edit1.text;
      close;
   end
   else
   if key=#27 then close;
end;

procedure Tpasswrd.FormCreate(Sender: TObject);
begin
   password:='';
end;

end.
```

MES.PAS – модуль формування повідомлення

```
unit Mes;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, QuickRpt, StdCtrls, ComCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    RichEdit1: TRichEdit;
    dlg1: TOpenDialog;
    Button3: TButton;
    Memo1: TMemo;
    Memo2: TMemo;
    Memo3: TMemo;
    Memo4: TMemo;
    Button2: TButton;
    Button4: TButton;
    sdlg: TSaveDialog;
    procedure Button1Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure FormResize(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
     fmode : integer;
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

function delspace(s : string):string;
begin
   while (LENGTH(S)>0) and (s[1]=' ') do delete (s,1,1);
   while (LENGTH(S)>0) and (s[length(s)]=' ') do delete (s,length(s),1);
   delspace:=s;
end;

procedure TForm1.Button1Click(Sender: TObject);
var
   x,y : integer;
   s : string;
   pozs : array [1..13] of integer;
   df : array [1..13] of shortstring;
procedure separate;
begin
   df[1]:=delspace(copy(s,pozs[1],pozs[2]-pozs[1]));
   df[2]:=delspace(copy(s,pozs[2],pozs[3]-pozs[2]));
   df[3]:=delspace(copy(s,pozs[3],pozs[4]-pozs[3]));
   df[4]:=delspace(copy(s,pozs[4],pozs[5]-pozs[4]));
   df[5]:=delspace(copy(s,pozs[5],pozs[6]-pozs[5]));
   df[6]:=delspace(copy(s,pozs[6],pozs[7]-pozs[6]));
   df[7]:=delspace(copy(s,pozs[7],pozs[8]-pozs[7]));
   df[8]:=delspace(copy(s,pozs[8],pozs[9]-pozs[8]));
   df[9]:=delspace(copy(s,pozs[9],pozs[10]-pozs[9]));
```

```
   df[10]:=delspace(copy(s,pozs[10],pozs[11]-pozs[10]));
   df[11]:=delspace(copy(s,pozs[11],pozs[12]-pozs[11]));
end;
begin
   if fmode=1 then
   begin
      if not dlg1.execute then exit;
      memo2.lines.loadfromfile(dlg1.filename);
   end
   else fmode:=1;
   richedit1.lines.clear;
   richedit1.lines.add(memo2.lines.strings[1]+'  '+memo2.lines.strings[3]);
   with memo2 do
   begin
      for   x:=0   to   5   do   if   (pos('Дата',lines.strings[x])<>0)   and
(pos('Повідомлення',lines.strings[x])<>0) then
      begin
         s:=lines.strings[x];
         break;
      end;

      for x:=7 to lines.count do
      begin
         s:=lines.strings[x];
         separate;

         memo1.lines.clear;
         memo3.lines.clear;
         memo4.lines.clear;
         memo1.lines.add(df[8]);
         memo3.lines.add(df[9]+' '+df[10]);
         memo4.lines.add(df[11]);
         s:=df[1];
         while length(s)<10 do s:=s+' ';
         s:=s+df[2];
         while length(s)<32 do s:=s+' ';
         s:=s+df[3];
         while length(s)<54 do s:=s+' ';
         s:=s+df[4];
         while length(s)<60 do s:=s+' ';

         if (length(df[5])>0) and (pos(',',df[5])=0) then df[5]:=df[5]+',';
         while length(df[5])<10 do df[5]:=' '+df[5];
         s:=s+df[5];
         while length(s)<72 do s:=s+' ';

         if (length(df[6])>0) and (pos(',',df[6])=0) then df[6]:=df[6]+',';
         while length(df[6])<10 do df[6]:=' '+df[6];
         s:=s+df[6];
         while length(s)<83 do s:=s+' ';

         if (length(df[7])>0) and (pos(',',df[7])=0) then df[7]:=df[7]+',';
         while length(df[7])<12 do df[7]:=' '+df[7];
         s:=s+df[7];
         while length(s)<97 do s:=s+' ';

         s:=s+memo1.lines.strings[0];
         while length(s)<115 do s:=s+' ';

         s:=s+memo3.lines.strings[0];
         while length(s)<132 do s:=s+' ';

         s:=s+memo4.lines.strings[0];
         richedit1.lines.add(s);

         y:=1;
         while     (memo1.lines.count>y)     or     (memo3.lines.count>y)     or
(memo4.lines.count>y) do
         begin
```

```
            s:='
';
            if (memo1.lines.count>y) then s:=s+memo1.lines.strings[y];
            while length(s)<115 do s:=s+' ';
            if (memo3.lines.count>y) then s:=s+memo3.lines.strings[y];
            while length(s)<132 do s:=s+' ';
            if (memo4.lines.count>y) then s:=s+memo4.lines.strings[y];
            richedit1.lines.add(s);
            inc (y);
         end;
         richedit1.lines.add('----------------------------------------------
------------------------------------------------------------------------------
----------------------');
      end
   end;
   with richedit1 do
   begin
      selstart:=0;
      sellength:=65535;
      selattributes.name:='courier';
      sellength:=0;
      try
         setfocus;
      except
      end;
   end;
end;


procedure TForm1.FormResize(Sender: TObject);
begin
   richedit1.width:=clientwidth;
   richedit1.height:=clientheight-button1.height;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
   if dlg1.execute then
   richedit1.lines.loadfromfile(dlg1.filename);
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
   if sdlg.execute then
   richedit1.lines.savetofile(sdlg.filename);
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
   fmode:=1;
end;

end.
```

RS_CODE.PAS – перешкодостійке кодування методом Ріда-Соломона

```
unit RS_CODE;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, XPMan;

type
  TForm1 = class(TForm)
    XPManifest1: TXPManifest;
    PC1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    Button1: TButton;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Memo1: TMemo;
    Memo2: TMemo;
    Label5: TLabel;
    Memo3: TMemo;
    Label6: TLabel;
    Memo4: TMemo;
    Memo5: TMemo;
    Memo6: TMemo;
    Label7: TLabel;
    Label1: TLabel;
    Button2: TButton;
    Label8: TLabel;
    procedure Button2Click(Sender: TObject);
    procedure kodhex;
    procedure binar;
    procedure kodhem;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  hex:array[1..2,1..8000] of integer;
  dlina:integer; //довжина тексту

implementation

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
kodhex;
binar;
kodhem;
end;
       //сам алгоритм кодування Ріда-Соломона
procedure tform1.kodhem;
var
s,bina,hem:string;
a,b,i,z,b1,b2,b3:integer;
begin
s:=memo1.Text;
hem:='';
```

```
z:=length(s);
a:=1;
while a<z do
begin
i:=0;
if s[a]='1' then i:=i xor 3;
if s[a+1]='1' then i:=i xor 5;
if s[a+2]='1' then i:=i xor 6;
if s[a+3]='1' then i:=i xor 7;
b:=i mod 8;
b3:=b div 4;
b:=b mod 4;
b2:=b div 2;
b1:=b mod 2;
bina:=inttostr(b1)+inttostr(b2)+s[a]+inttostr(b3)+s[a+1]+s[a+2]+s[a+3];
hem:=hem+bina;
a:=a+4;
end;
memo2.Text:=hem;
memo3.Text:=hem;
end;
       //Перетворення у двійковий вигляд
procedure tform1.binar;
var
a,b,temp,t1:integer;
bin:string;
begin
bin:='';
for a:=1 to dlina do
begin
for b:=1 to 2 do
begin
temp:=hex[b,a] div 8;
bin:=bin+inttostr(temp);
t1:=hex[b,a] mod 8;
temp:=t1 div 4;
bin:=bin+inttostr(temp);
t1:=t1 mod 4;
temp:=t1 div 2;
bin:=bin+inttostr(temp);
temp:=t1 mod 2;
bin:=bin+inttostr(temp);
end;
end;
memo1.Text:=bin;
end;
       //Перетворення у шістнадцятковий вигляд
procedure tform1.kodhex;
var
s,h,h1,h2:string;
b,i:integer;
begin
s:=memo6.Text;
dlina:=length(s);
if dlina=0 then exit;
h:='';
for b:=1 to dlina do
begin
i:=ord(s[b]);
hex[1,b]:=i div 16;
h1:=inttostr(hex[1,b]);
case hex[1,b] of
10:h1:='A';
11:h1:='B';
12:h1:='C';
13:h1:='D';
14:h1:='E';
15:h1:='F';
end;
```

```
hex[2,b]:=i-(hex[1,b]*16);
h2:=inttostr(hex[2,b]);
case hex[2,b] of
10:h2:='A';
11:h2:='B';
12:h2:='C';
13:h2:='D';
14:h2:='E';
15:h2:='F';
end;
h:=h+h1+h2+',';
end;
delete(h,length(h),1);
memo5.Text:=h;
end;
//Підпрограмма визначення кількості помилок та їх виправлення, якщо вони є
procedure TForm1.Button2Click(Sender: TObject);
var
s:string;
a,b,i,z,f,osh:integer;
begin
s:=memo3.Text;
z:=length(s);
a:=1;
osh:=0;
while a<z do
begin
i:=0;
for f:=0 to 6 do if s[a+f]='1' then i:=i xor (f+1);
i:=i mod 8;
if i<>0 then
begin
inc(osh);
if s[a+i-1]='0' then s[a+i-1]:='1' else s[a+i-1]:='0';
end;
a:=a+7;
end;
label8.Caption:='Знайдено помилок    '+inttostr(osh)+' шт.';
memo4.Text:=s;
end;

end.
```

AES_FOR_KEY.PAS – шифрування стегоключа алгоритмом AES_FOR_KEY

```
unit AES_FOR_KEY;

interface

Uses Windows, Classes, SysUtils, Math, Dialogs;

Type
  TBitString = Array of Boolean;
  PBitString = ^TBitString;

  TSplitKeyParts = record
    C:TBitString;
    D:TBitString;
    end;
  TSplitKey = Array[0..16]Of TSplitKeyParts;

  TConcatKey = Array[0..15]Of TBitString;

  TIPKeyParts = record
    L:TBitString;
    R:TBitString;
    end;
  TIPKey = Array[0..16]OF TIPKeyParts;

Const
AES_FOR_KEY_PC1:Array[0..55] Of Byte = (57,49,41,33,25,17,9,
                                1,58,50,42,34,26,18,
                                10,2,59,51,43,35,27,
                                19,11,3,60,52,44,36,
                                63,55,47,39,31,23,15,
                                7,62,54,46,38,30,22,
                                14,6,61,53,45,37,29,
                                21,13,5,28,20,12,4);

AES_FOR_KEY_PC2:Array[0..47] Of Byte = (14,17,11,24,1,5,
                                3,28,15,6,21,10,
                                23,19,12,4,26,8,
                                16,7,27,20,13,2,
                                41,52,31,37,47,55,
                                30,40,51,45,33,48,
                                44,49,39,56,34,53,
                                46,42,50,36,29,32);

AES_FOR_KEY_IP:Array[0..63] Of Byte = (58,50,42,34,26,18,10,2,
                               60,52,44,36,28,20,12,4,
                               62,54,46,38,30,22,14,6,
                               64,56,48,40,32,24,16,8,
                               57,49,41,33,25,17,9,1,
                               59,51,43,35,27,19,11,3,
                               61,53,45,37,29,21,13,5,
                               63,55,47,39,31,23,15,7);

AES_FOR_KEY_E:Array[0..47] Of Byte = (32,1,2,3,4,5,
                              4,5,6,7,8,9,
                              8,9,10,11,12,13,
                              12,13,14,15,16,17,
                              16,17,18,19,20,21,
                              20,21,22,23,24,25,
                              24,25,26,27,28,29,
                              28,29,30,31,32,1);

S_BOXES:Array[0..7,0..3,0..15]Of Byte = (
((14,04,13,01,02,15,11,08,03,10,06,12,05,09,00,07),
  (00,15,07,04,14,02,13,01,10,06,12,11,09,05,03,08),
```

```
  (04,01,14,08,13,06,02,11,15,12,09,07,03,10,05,00),
  (15,12,08,02,04,09,01,07,05,11,03,14,10,00,06,13)),

((15,01,08,14,06,11,03,04,09,07,02,13,12,00,05,10),
      (03,13,04,07,15,02,08,14,12,00,01,10,06,09,11,05),
      (00,14,07,11,10,04,13,01,05,08,12,06,09,03,02,15),
      (13,08,10,01,03,15,04,02,11,06,07,12,00,05,14,09)),

((10,00,09,14,06,03,15,05,01,13,12,07,11,04,02,08),
      (13,07,00,09,03,04,06,10,02,08,05,14,12,11,15,01),
      (13,06,04,09,08,15,03,00,11,01,02,12,05,10,14,07),
      (01,10,13,00,06,09,08,07,04,15,14,03,11,05,02,12)),

((07,13,14,03,00,06,09,10,01,02,08,05,11,12,04,15),
      (13,08,11,05,06,15,00,03,04,07,02,12,01,10,14,09),
      (10,06,09,00,12,11,07,13,15,01,03,14,05,02,08,04),
      (13,15,00,06,10,01,13,08,09,04,05,11,12,07,02,14)),

((02,12,04,01,07,10,11,06,08,05,03,15,13,00,14,09),
      (14,11,02,12,04,07,13,01,05,00,15,10,03,08,09,06),
      (04,02,01,11,10,13,07,08,15,09,12,05,06,03,00,14),
      (11,08,12,07,01,14,02,13,06,15,00,09,10,04,05,03)),

((12,01,10,15,09,02,06,08,00,13,03,04,14,07,05,11),
      (10,15,04,02,07,12,09,05,06,01,13,14,00,11,03,08),
      (09,14,15,05,02,08,12,03,07,00,04,10,01,13,11,06),
      (04,03,02,12,09,05,15,10,11,14,01,04,06,00,08,13)),

((04,11,02,14,15,00,08,13,03,12,09,07,05,10,06,01),
      (13,00,11,07,04,09,01,10,14,03,05,12,02,15,08,06),
      (01,04,11,13,12,03,07,14,10,15,06,08,00,05,09,02),
      (06,11,13,08,01,04,10,07,09,05,00,15,14,02,03,12)),

((13,02,08,04,06,15,11,01,10,09,03,14,05,00,12,07),
      (01,15,13,08,10,03,07,04,12,05,06,11,00,14,09,02),
      (07,11,04,01,09,12,14,02,00,06,10,13,15,03,05,08),
      (02,01,14,07,04,10,08,13,15,12,09,00,03,05,06,11))
);

AES_FOR_KEY_P:Array[0..31] Of Byte = (16,7,20,21,
                              29,12,28,17,
                              1,15,23,26,
                              5,18,31,10,
                              2,8,24,14,
                              32,27,3,9,
                              19,13,30,6,
                              22,11,4,25);

AES_FOR_KEY_REVERSE_IP:Array[0..63] Of Byte = (40,8,48,16,56,24,64,32,
             39,7,47,15,55,23,63,31,
             38,6,46,14,54,22,62,30,
             37,5,45,13,53,21,61,29,
             36,4,44,12,52,20,60,28,
             35,3,43,11,51,19,59,27,
             34,2,42,10,50,18,58,26,
             33,1,41,9,49,17,57,25);

AES_FOR_KEY_LSH:Array[0..15] Of Byte = (1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1);

Function BinToInt(S:TBitString):Integer;
Function IntToBin(N:Integer;Precision:Integer=8):TBitString;

Function BinToStr(Bits:TBitString):String;
Function StrToBin(S:String):TBitString;

Function AnsiStrToBin(S:String; Zeroes:Boolean=True):TBitString;
Function BinToAnsiStr(Bits:TBitString):String;
```

```
Procedure CopyBits(Var AES_FOR_KEYt:TBitString; Source:TBitString;
NBits:Integer);
Function ConcatBits(Bits:Array Of TBitString):TBitString;

Function AES_FOR_KEYEncode(S,Key:String):TBitString;
Function AES_FOR_KEYDecode(S,Key:String):TBitString;

Function GetPermutedKey(Key:TBitString):TBitString;
Function GetPermutedKey2(Key:TBitString):TBitString;

Function GetSplitKey(Key:TBitString):TSplitKey;
Function GetConcatKey(Key:TSplitKey):TConcatKey;
Function GetIPKey(M:TBitString; ConcatKey:TConcatKey):TIPKey;
Function Get(R,K:TBitString):TBitString;
Function GetSBox(Index:Integer; T:TBitString):TBitString;
Function GetReverseIP(RL:TBitString):TBitString;
Procedure ReverseSubKeys(Var Keys:TConcatKey);

implementation

Function ConcatBits(Bits:Array Of TBitString):TBitString;
Var
I,C:Integer;
Begin
SetLength(Result,0);
For C:=0 To Length(Bits)-1 Do
  Begin
  SetLength(Result,Length(Result)+Length(Bits[C]));
  For I:=0 To Length(Bits[C])-1 Do
    Result[Length(Result)-Length(Bits[C])+I]:=Bits[C][I];
  End;
End;

Procedure CopyBits(Var AES_FOR_KEYt:TBitString; Source:TBitString;
NBits:Integer);
Var
I:Integer;
Begin
SetLength(AES_FOR_KEYt,NBits);
For I:=0 To NBits-1 Do
  AES_FOR_KEYt[I]:=Source[I];
End;

Function BinToInt(S: TBitString): Integer;
Var
L,I:Integer;
Begin
Result:=0;
L:=Length(S);
IF L=0 Then
  Raise EConvertError.Create(' Бітовий рядок довжини нуль ');
For I:= L-1 DownTo 0 Do
  Result:=Result+Ord(S[I])*Trunc(Power(2, L-I-1));
End;

Function IntToBin(N:Integer; Precision:Integer):TBitString;
Var
BitList:TList;
Bit:PBoolean;
Begin
SetLength(Result,0);
BitList:=TList.Create;
While N>0 Do
  Begin
  New(Bit);
  Bit^:=Boolean(N mod 2);
  BitList.Insert(0,Bit);
  N:=N div 2;
  End;
```

```
While BitList.Count<Precision Do
  Begin
  New(Bit);
  Bit^:=False;
  BitList.Insert(0,Bit);
  End;
For N:=0 To BitList.Count-1 Do
  Begin
  SetLength(Result,N+1);
  Bit:=BitList.Items[N];
  Result[N]:=Bit^;
  Dispose(Bit);
  End;
BitList.Free;
end;

Function AnsiStrToBin(S: String; Zeroes:Boolean):TBitString;
Var
Temp,B:TBitString;
L,I,J:Integer;
Begin
L:=0;
SetLength(Result,L);
SetLength(Temp,L);
SetLength(B,0);
For I:=1 To Length(S) Do
  Begin
  B:=IntToBin(Ord(S[I]));
  L:=L+Length(B);
  SetLength(Temp,L);
  For J:=0 To Length(B)-1 Do
    Temp[Length(Temp)-Length(B)+J]:=B[J];
  End;
Result:=Temp;
End;

Function BinToStr(Bits:TBitString):String;
Var
I,L:Integer;
Begin
Result:='';
L:=Length(Bits);
IF L=0 Then
  Raise EConvertError.Create(' Бітовий рядок довжини нуль ');
For I:=0 To L-1 Do
  IF Bits[I] Then Result:=Result+'1'
  Else Result:=Result+'0';
End;

Function StrToBin(S:String):TBitString;
Var
I:Integer;
Begin
SetLength(Result,0);
For I:=1 To Length(S) Do
  Begin
  IF (S[I]<>'1')And(S[I]<>'0') Then
    Raise EConvertError.Create(S+' помилковий двійковий рядок');
  SetLength(Result,I);
  Result[ I-1]:=Boolean(StrToInt(S[I]));
  End;
End;

Function BinToAnsiStr(Bits:TBitString):String;
Var
I:Integer;
B:TBitString;
Begin
Result:='';
```

```
SetLength(B,8);
I:=0;
While I<=Length(Bits)-8 Do
  Begin
  CopyMemory(B,Ptr(Integer(Bits)+I),8);
  Result:=Result+Char(BinToInt(B));
  Inc(I,8);
  End;
End;

Function GetPermutedKey(Key:TBitString):TBitString;
Var
I:Integer;
Begin
SetLength(Result,Length(AES_FOR_KEY_PC1));
For I:=0 To Length(AES_FOR_KEY_PC1)-1 Do
  Result[I]:=Key[AES_FOR_KEY_PC1[I]-1];
End;

Function GetPermutedKey2(Key:TBitString):TBitString;
Var
I:Integer;
Begin
SetLength(Result,Length(AES_FOR_KEY_PC2));
For I:=0 To Length(AES_FOR_KEY_PC2)-1 Do
  Result[I]:=Key[AES_FOR_KEY_PC2[I]-1];
End;

Function GetSplitKey(Key:TBitString):TSplitKey;
  Function LeftShift(Key:TBitString; N:Integer):TBitString;
  Var
  I,J:Integer;
  Temp:TBitString;
  Begin
  SetLength(Result,28);
  SetLength(Temp,28);
  For I:=0 To 27 Do
    Temp[I]:=Key[I];
  For J:=1 To N Do
    Begin
    For I:=1 To 27 Do
      Result[ I-1]:=Temp[I];
    Result[27]:=Temp[0];
    For I:=0 To 27 Do
      Temp[I]:=Result[I];
    End;
  End;
Var
I,J:Integer;
Begin
For J:=1 To 16 Do
  Begin
  SetLength(Result[J].C,28);
  SetLength(Result[J].D,28);
  End;
CopyBits(Result[0].C,Key,28);
CopyBits(Result[0].D,TBitString(Integer(Key)+28),28);
For I:=1 To 16 Do
  Begin
  Result[I].C:=LeftShift(Result[ I-1].C,AES_FOR_KEY_LSH[ I-1]);
  Result[I].D:=LeftShift(Result[ I-1].D,AES_FOR_KEY_LSH[ I-1]);
  End;
End;

Function GetConcatKey(Key:TSplitKey):TConcatKey;
Var
I:Integer;
Temp:TBitString;
Begin
```

```
For I:=0 To 15 Do
  Begin
  SetLength(Result[I],56);
  Temp:=ConcatBits([Key[I+1].C,Key[I+1].D]);
  Result[I]:=GetPermutedKey2(Temp);
  End;
End;

Function GetIPKey(M:TBitString; ConcatKey:TConcatKey):TIPKey;
Var
I,J:Integer;
IP, F:TBitString;
Begin
For I:=0 To 16 Do
  Begin
  SetLength(Result[I].L,32);
  SetLength(Result[I].R,32);
  End;

SetLength(IP,64);
For I:=0 To Length(AES_FOR_KEY_IP)-1 Do
  IP[I]:=M[AES_FOR_KEY_IP[I]-1];

For I:=0 To 31 Do
  Result[0].L[I]:=IP[I];
For I:=32 To 63 Do
  Result[0].R[ I-32]:=IP[I];

For I:=1 To 16 Do
  Begin
  Result[I].L:=Result[ I-1].R;
  F:=Get(Result[ I-1].R,ConcatKey[ I-1]);
  For J:=0 To 31 Do
    Result[I].R[J]:=Result[ I-1].L[J] XOR F[J];
  End;
End;

Function Get(R,K:TBitString):TBitString;
Var
I,J:Integer;
S,E,KE,F,T:TBitString;
Begin
SetLength(E,48);
For I:=0 To 47 Do
  E[I]:=R[AES_FOR_KEY_E[I]-1];

SetLength(KE,48);
For I:=0 To 47 Do
  KE[I]:=K[I] XOR E[I];

SetLength(T,6);
SetLength(F,0);
SetLength(S,4);
I:=0;
While I<48 Do
  Begin
  For J:=0 To 6 Do
    T[J]:=KE[J+I];
  S:=GetSBox(I div 6,T);
  F:=ConcatBits([F,S]);
  I:=I+6;
  End;
SetLength(Result,32);
For I:=0 To 31 Do
  Result[I]:=F[AES_FOR_KEY_P[I]-1];
End;

Function GetSBox(Index:Integer; T:TBitString):TBitString;
Var
```

```
Val,Row,Col:Integer;
Temp:TBitString;
Begin
SetLength(Result,4);
SetLength(Temp,2);
Temp[0]:=T[0];
Temp[1]:=T[5];
Row:=BinToInt(Temp);
SetLength(Temp,4);
CopyBits(Temp,TBitString(@T[1]),4);
Col:=BinToInt(Temp);
Val:=S_BOXES[Index,Row,Col];
SetLength(Result,4);
Result:=IntToBin(Val,4);
End;

Function GetReverseIP(RL:TBitString):TBitString;
Var
I:Integer;
Begin
SetLength(Result,64);
For I:=0 To Length(AES_FOR_KEY_REVERSE_IP)-1 Do
  Result[I]:=RL[AES_FOR_KEY_REVERSE_IP[I]-1];
End;

Procedure ReverseSubKeys(Var Keys:TConcatKey);
Var
I,L:Integer;
T:TBitString;
Begin
SetLength(T,48);
L:=Length(Keys);
For I:=0 To ( L-1)Div 2 Do
  Begin
  T:=Keys[I];
  Keys[I]:=Keys[( L-I)-1];
  Keys[( L-I)-1]:=T;
  End;
End;

Function AES_FOR_KEYEncode(S,Key:String):TBitString;
Var
I:Integer;
K:TBitString;
M:TBitString;
RL:TBitString;
Kplus:TBitString;
SplitKey:TSplitKey;
ConcatKey:TConcatKey;
IPKey:TIPKey;
Begin
K:=AnsiStrToBin(Key);
Kplus:=GetPermutedKey(K);
SplitKey:=GetSplitKey(Kplus);
ConcatKey:=GetConcatKey(SplitKey);
M:=AnsiStrToBin(S);
IPKey:=GetIPKey(M,ConcatKey);
SetLength(RL,64);
For I:=0 To 31 Do
  Begin
  RL[I]:=IPKey[16].R[I];
  RL[I+32]:=IPKey[16].L[I];
  End;
RL:=GetReverseIP(RL);
Result:=RL;
End;

Function AES_FOR_KEYDecode(S,Key:String):TBitString;
Var
```

```
I:Integer;
K:TBitString;
M:TBitString;
RL:TBitString;
Kplus:TBitString;
SplitKey:TSplitKey;
ConcatKey:TConcatKey;
IPKey:TIPKey;
Begin
K:=AnsiStrToBin(Key);
Kplus:=GetPermutedKey(K);
SplitKey:=GetSplitKey(Kplus);
ConcatKey:=GetConcatKey(SplitKey);
ReverseSubKeys(ConcatKey);
M:=AnsiStrToBin(S);
IPKey:=GetIPKey(M,ConcatKey);
SetLength(RL,64);
For I:=0 To 31 Do
  Begin
  RL[I]:=IPKey[16].R[I];
  RL[I+32]:=IPKey[16].L[I];
  End;
RL:=GetReverseIP(RL);
Result:=RL;
End;

end.
```

MAIN_STEGOGRAPHY_FOR_NET.PAS – основна программа

```
unit main_STEGOGRAPHY_FOR_NET;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls, Menus, _STEGOGRAPHY_FOR_NET, about;

type

  T_STEGOGRAPHY_FOR_NET = class(TForm)
    Button1: TButton;
    loadbmp: TOpenDialog;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    ScrollBox1: TScrollBox;
    Image1: TImage;
    Button5: TButton;
    PopupMenu1: TPopupMenu;
    N1: TMenuItem;
    N3: TMenuItem;
    N5: TMenuItem;
    N6: TMenuItem;
    Main_STEGOGRAPHY_FOR_NETMenu1: TMain_STEGOGRAPHY_FOR_NETMenu;
    Loadfile1: TMenuItem;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure FormResize(Sender: TObject);
    procedure N6Click(Sender: TObject);
  private
     pt : TBuffer_;
     loaded_ : boolean;
  end;

var
  _STEGOGRAPHY_FOR_NET: T_STEGOGRAPHY_FOR_NET;

implementation

uses password__STEGOGRAPHY_FOR_NET, RS_CODE;

{$R *.DFM}

// завантаження зображення
procedure T_STEGOGRAPHY_FOR_NET.Button1Click(Sender: TObject);
begin
    if not loadbmp.execute then exit;
image1.picture.bitmap.loadfromfile(loadbmp.filename);
// перевірка формату малюнка. Треба 24-бітний.
    if image1.picture.bitmap.pixelformat<>pf24bit then
// Формат малюнка не підходить. Запит на перетворення формату
     if application.messagebox('Можлива робота лише з 24-бітними зображеннями.
Конвертувати?','',$11)=1 then
        image1.picture.bitmap.pixelformat:=pf24bit;
     maxcol:=((image1.picture.bitmap.width) * 3);
// максимальний об’єм даних, які можна помістити в зображення
maxlen:=((maxcol*image1.picture.bitmap.height) div 8)-25;
    if maxlen<=0 then
    begin
       maxlen:=0;
       loaded_:=false;
    end
    else loaded_:=true;
```

```
   checkbmp;
   form2.label6.caption:=inttostr(maxlen);
   form2.label3.caption:=inttostr(maxlen);
   estlen:=maxlen;
end;

procedure T_STEGOGRAPHY_FOR_NET.FormCreate(Sender: TObject);
begin
   loaded_:=false;
end;

procedure T_STEGOGRAPHY_FOR_NET.Button4Click(Sender: TObject);
begin
   if not loaded_ then exit;
   form2.loaddir;    // Процедура читання вбудованої інформації
   form2.showmodal;
end;

procedure T_STEGOGRAPHY_FOR_NET.Button5Click(Sender: TObject);
begin
   passwrd.edit1.text:=passwrd.password;     // Запит пароля
   passwrd.showmodal;
end;

procedure T_STEGOGRAPHY_FOR_NET.FormResize(Sender: TObject);
begin
   scrollbox1.width:=clientwidth;
   scrollbox1.height:=clientheight-scrollbox1.top;
end;

procedure T_STEGOGRAPHY_FOR_NET.N6Click(Sender: TObject);
begin
   form1.show;
end;

end.
```