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
“____” _____ 2024 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за першим (бакалаврським) рівнем вищої освіти
на тему
“Програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії”

Виконав здобувач вищої освіти
IV курсу, групи КБ-20
ОПП «Кібербезпека»
спеціальності 125 «Кібербезпека»
_____ Сосна О.С.
«____» _____ 2024 р.

Керівник проекту
доктор технічних наук, професор
_____ Смірнов О.А.
«____» _____ 2024 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 2024 року

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

Сосні Олександру Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи *Програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії*

2. Керівник роботи *Смірнов Олексій Анатолійович, докт. техн. наук, професор*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 135-02 від 01.04.2024 року

3. Строк подання студентом роботи до захисту *23.05.2024 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою роботи є розробка програмного забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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 2024 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	10.03.2024 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2024 р.	
3.	Розробка моделі компонента	20.03.2024 р.	
4.	Розробка структур даних	25.03.2024 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2024 р.	
6.	Програмування алгоритмів	10.04.2024 р.	
7.	Оформлення ПЗ	17.04.2024 р.	
8.	Попередній захист роботи	23.05.2024 р.	

Дата видачі завдання
« 17 » січня 2024 р.

Підпис керівника
_____ Смірнов О.А.
(прізвище та ініціали)

Завдання прийнято до виконання
« 17 » січня 2024 р.

Підпис здобувача
_____ Сосна О.С.
(прізвище та ініціали)

АНОТАЦІЯ

Сосна О.С. Програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії. 125 Кібербезпека. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

В даній випускній кваліфікаційній роботі за першим (бакалаврським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Метою розробки є програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Результат роботи – програмна реалізація системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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 з ОС Windows 10/11.

Програму розроблено в середовищі Delphi 10.4 Sydney.

Ключові слова: кібербезпека, стеганографія

ABSTRACT

Sosna O.S. Software of the cyber security system of data marking of micropayment photobanks using steganography methods. 125 Cyber security. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this graduation thesis for the first (bachelor) level of higher education, software is developed, which is intended for the cyber security system of data marking of micropayment photobanks using steganography methods.

The purpose of the development is the software of the cyber security system for marking the data of micropayment photobanks using steganography methods.

The result of the work is the software implementation of the cyber security system for marking the data of micropayment photobanks using steganography methods.

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 a PC with Windows 10/11 OS.

The program was developed in the Delphi 10.4 Sydney environment.

Keywords: cyber security, steganography

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ВДТ	–	відео-дисплейні термінали
ЕОМ	–	електронно-обчислювальна машина
ЕЦП	–	електронний цифровий підпис
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. Перспективним напрямком для захисту авторських прав на дані мікропейментових фотобанків є цифрова (комп'ютерна) стеганографія. В той же час, у зв'язку з бурхливим розвитком сучасних інформаційних технологій, зростанням обсягів інформації, яка передається і обробляється в комп'ютерних системах та мережах, появою нових інформаційних послуг та сервісів, існуючий на сьогоднішній день математичний апарат стеганографічного захисту інформації, не дозволяє реалізувати забезпечення підвищених вимог. Таким чином, збільшення обсягів даних, які обробляються і передаються в комп'ютерних системах та мережах, підвищення ймовірнісно-часових вимог до безпеки, вірогідності і пропускної здатності стеганографічних каналів передачі даних, поява нових загроз авторським правам на дані мікропейментових фотобанків обумовлюють об'єктивно існуюче протиріччя, на вирішення якого і спрямована мета даної роботи.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

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

– Огляд існуючих систем маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

– Дослідження системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

– Програмна реалізація системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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

Система призначена для реалізації програмного забезпечення маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Стеганографія – це приховання самого факту передачі інформації. Це принципово відрізняється від криптографії. У випадку з криптографічними перетвореннями факт приховання інформації очевидний. Тобто, коли А передає інформацію до Б, то В знаєш, що інформація є секретна, проте не має (в кращому випадку) алгоритмів її розшифрування, якщо вона потрапить йому до рук.

Стеганографічні перетворення повідомлення дозволяють приховати від зловмисника сам факт передачі секретної інформації. Тобто, В, можливо і помітить обмін інформацією між А і Б, але не вбачатиме в ній нічого цінного.

Приховання факту передачі інформації зменшує ризики того, що секретна або конфедеційна інформація потрапить до зловмисника, а навіть якщо і потрапить він може не побачити в ній нічого цінного.

У випадку з стегоповідомленням, зловмисник навіть не матиме мотивації для таких дій, тому що стегоповідомлення не привертає до себе уваги.

Головний принцип стеганографії полягає у тому, щоб приховати конфеденційну інформацію всередині відкритою, як правило вседоступної інформації. Тобто один тип інформації (текст, зображення, аудіо і т.д.) поміщається всередину іншої інформації (текст, зображення, аудіо і т.д.). Таким чином контейнер (інформація, що приховує у собі стегоповідомлення) виглядає безобідно.

Найпоширенішим з методів, який читач може зустріти в Інтернеті є приховання тексту в зображенні.

1.2 Область застосування

Областю застосування системи є мікропейментові фотобанки з застосуванням методів стеганографії.

Термін "Stock Photography" можна перекласти як "Архівна фотографія" (Stock – запас; мати в наявності, асортимент, так само будь-яка сукупність об'єктів, загальних ознак, що характеризуються набором). Тобто, стоковою фотографією може вважатися будь-який знімок, розміщений у фото-банку. Але все ж таки, можна дати деяке визначення – це технічна фотографія, що не створена під певний проект, а передає собою якусь ідею, або ж сама є джерелом ідеї. Ці фотографії розміщені у фото-банках, або фотоагенствах або фотостоках. Загалом – це каталог знімків найрізноманітніших тематик.

Мікропейментові фотобанки

Особливість даних банків полягає в тому, що вартість кожної фотографії, яка продається, через стандартний тип ліцензування є дуже маленькою, як правило один або декілька доларів. З них фотографові дістається 20%-80%. Але за рахунок великої кількості продажів досягається відчутний дохід. Хоча типів ліцензій існує декілька, все ж таки левова частка продажів відбувається саме за стандартною. Одна і та ж фотографія може продаватися десятки разів на місяць, іноді сотні, хоча може й не продаватися взагалі – багато що залежить від змісту знімка (про це нижче). Одна важлива деталь – завантаживши в такий банк зображення, Ви не втрачаєте на нього авторські права, і можете з таким же успіхом розміщувати це ж саме зображення в інших аналогічних фото-банках, отримуючи прибуток і звідти.

Купуючи знімок за розширеною ліцензією, покупець має право використовувати зображення у виданнях, що дозволяє використовувати зображення в комерційних цілях, де відтворення цих фотоматеріалів пов'язане з фінансовим прибутком: у рекламній продукції, корпоративних брошурах, на упакуваннях, веб-сайтах, в мультимедійних проектах і таке інше. Купівля

зображення за такою ліцензією зазвичай обходиться покупцеві в суму від 20 до 50 доларів. Ліцензія Right Managed має на увазі видалення цієї картки з інших агентств після покупки. Або й взагалі, продаж здійснюється тільки через один банк. Ціна за фотокартку може складати 300-500 доларів. І останній вид ліцензії Exclusive Buyout . викуп усіх прав на зображення. Тобто фотограф позбавляється авторських прав і правовласником стає покупець. Але заплатити він винен за таке задоволення близько 2000-5000 доларів.

Щодня з'являється на світ величезна кількість проектів, які повинні бути проілюстровані хорошими знімками, – веб-сайти, корпоративні буклети, періодичні видання і рекламна продукція. І як Ви розумієте, не завжди є можливим наймати професійного фотографа (через брак часу, фінансів, розуміння що саме потрібне і т.і.).

Тому зручним в усіх відношеннях рішенням, є використання стокової фотографії, купленої у фотобанку. За порівняно невеликі гроші та заощаджуючи при цьому час, замовник отримує фотозображення хорошої якості (часто навіть кращої, ніж у деяких найманих фотографів) з дотриманням усіх авторських прав і отриманням усіх належних документів. Фотограф отримує матеріальну винагороду і моральне задоволення. Усе автоматизовано і відбувається практично вмить – ні для кого ніякої мороки. Згадаємо і власників фотобанків – вони отримують комісійні.

Відправляти у фотобанки можна тільки зроблені Вами фотографії. Якщо на знімках є обличчя людей – потрібно прикласти розписку моделі, тобто, людина повинна дозволити продавати фото зі своїм зображенням. Так само розписка потрібна і для деяких видів приватної власності, наприклад картин, предметів мистецтва і деяких будівель. Зразок розписки можна викачати на будь-якому фотостоці. Якість знімків повинна бути достатньо високою, отже потрібно мати не обов'язково професійну камеру, але не погану. І звичайно ж потрібно розуміти, що таке якісна фотографія. Шуми, неправильна експозиція, погане фокусування виключаються. За всім цим стежать інспектори, перевіряючи

фотографії після завантаження, це може займати декілька днів. Мінімальний розмір фотографій, що приймаються, у всіх фотостоках різний, але перевага надається знімкам принаймні 4 мегапікселів.

Що б ваші фотографії продавалися, окрім хорошої якості потрібне дещо ще. Як вже писалося, фотографії купують для якихось певних проектів. Роблять це не абстрактні істоти, а прості люди зі своїми запитами, яким потрібний певний сюжет або об'єкт для цих самих проектів. Отже, сам факт створення фотографії не дає гарантії що її буде продано велику кількість разів.

Дуже затребуваними є сюжетні фотографії, де задіяні люди. Особливо популярними є сфера охорони здоров'я (лікар оглядає пацієнта), діловодство, спорт і різні персонажі у дії. Об’єкти, особливо на білому фоні, рідкісні й раритетні речі, мода, свята і події в різні пори року, їжа і напої (так само ті, де в композицію включені люди) і так далі. Грубо кажучи знімок бізнесмена що підписує папери куплять сотні разів, пейзаж знятий з вікна – можливо декілька. Хоча завжди бувають винятки.

Не варто відправляти знімки образотворчого мистецтва, по-перше ситуація з дотриманням авторських прав, по-друге це все ж таки не галерея мистецтв. Фотографії сімейних улюбленців, включаючи собак котів, тарганів і мух, навіть якщо вони вийшли просто чудовими (краще дарувати їх своїй бабусі, в рамочці). Зображення квітів, заходів, і природа як така, теж не затребувана. Будь-який фотобанк переобтяжений фотографіями такого типу. Якщо вони дуже хороші, і їх все ж таки приймуть, то вони не матимуть попиту. Розуміння того, що потрібне приходить через певний час роботи з фотобанками.

Величезна перевага для фотографів у роботі з фотобанками в тому, що знімки відправляються через інтернет, поодинці або по декілька штук, а не на дисках поштою. Також, інші операції здійснюються через веб-сервер-інтерфейс, тобто не потрібно ні з ким контактувати – ні по телефону, ні поштою, ніяким іншим способом. Все, що потрібно мати – портфоліо фотографій або бажання спробувати себе в ролі стокового фотографа і, доступ до інтернету, бажано

швидкісний, – фотографії повинні бути хорошої якості, а значить великі за обсягом.

Отримувати свої гроші можна після накопичення певної суми – банківським переказом, чеком поштою, або ж через системи електронних платежів PayPal і Moneybookers.

За кордоном деякі фотографи обирають роботу з фотобанками, як основний вид діяльності, за який отримують непогані гонорари. З'явилася ця професія і у нас – стоковий фотограф. Будь-хто може спробувати себе в цій ролі, основна вимога – вміння робити хороші знімки.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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

Фотостоків за останні кілька років виникла величезне кількість, в тому числі і у нас в Україні їх вже більше десятка, однак новим фотостоках з базою в декілька тисяч зображень важко змагатися з монстрами, що розташовують більше мільйона різних зображень. Новачкам НЕ рекомендується працювати відразу з усіма фотостоками – у вас не вистачить ні сил, ні часу. Раджу працювати лише з перевіреними фотостоками, які будуть справедливо і чесно продавати Ваші фото і розраховуватись з Вами. Ось список найбільш успішних фотостоків:

Shutterstock (США)

Це один із самих великих фотобанків який продає стокові зображення по абонементу (у банку понад 2,5 млн. растрових і векторних зображень і понад 35 тис. відеороликів). Приймає фотографії розміром не менше4 мегапікселів, також векторні, флеш і відеоматеріали. Можна завантажувати безкоштовні зображення.

Розцінки для клієнтів: 249 дол коштує абонемент на місяць, відеоролики продаються окремо, від 39 до 199 дол за штуку. Винагорода авторові: 0,25-0,30 дол за RF-ліценізії, 20 дол за ERF-ліцензії, до 10 дол за реалізацію на партнерських веб-сайтах, 30% з реалізації відеороликів, мінімальний платіж – 75 дол, щомісяця. Особливості при реєстрації: форма реєстрації на англійській мові, потрібна фотографія закордонного паспорта і 10 тестових робіт.

iStockphoto (Canada)

Теж найстаріший фотобанк, реалізовує продажу лише поштучно. На даний момент у банку Istockphoto понад 2 млн. зображень. Фотостік знаменитий надзвичайно суворими вимогами як до змісту, так і до якості робіт авторів.

Приймає: фотографії розміром більше 1600x1200, векторні зображення, відео-і флеш-ролики.

Розцінки для клієнтів: від 1 до 40 дол за RF-і до 500 дол за ERF-ліцензії, відеоматеріали – 20-40 дол

Винагорода авторові: 20 – 40%, виходячи з рангу і ексклюзивності фотохудожника, мінімальний платіж – 100 дол за вимогою.

Особливості при реєстрації: іспит зі знання правил сервісу, фотографія паспорта та 3 тестові роботи.

Stockxpert

Зростаючий і надзвичайно зручний в роботі фотостік. Основна перевага полягає у швидкості приймання а також у тому, що атрибутувати зображення дозволено після схвалення інспектором, іншими словами ті фотографії, які вже прийняті в банк. Продають зображення якпоштучно і по абонементу.

Приймає: фотографії розміром більше 800x600, також векторні зображення.

Розцінки для клієнтів: від 1 до 4 дол по RF ліцензії, від 25 до 100 дол за ERF ліцензії.

Винагорода авторам: 50% від суми продажу, мінімальний платіж – 50 дол

Особливості при реєстрації: 5 тестових робіт.

Fotolia

Високоперспективній фотоосток з російськими працівниками у штаті фірми. Це молодий європейський фотостік, грунтовно зайняв тверду позицію між лідерів цієї індустрії. У базі Fotolia міститься понад 2,5 млн. зображень, які продаються поштучно.

Зареєструвавшись, Ви отримаєте можливість викачувати величезна кількість безкоштовних фотографій.

Приймає: фотографії розміром більше 1600×1200.

Розцінки для клієнтів: від 1 до 80 дол за RF-ліцензії, до 200 дол за ERF-ліцензії, від 100 до 2000 дол за реалізацію прав на роботу.

Винагорода авторам: від 33 до 80% виходячи з рангу і ексклюзивності фотохудожника, мінімальний платіж – 50 дол, якщоВи замовите менш зазначеної суми, то доведеться заплатити комісію – 1 дол

Особливості при реєстрації: немає.

Dreamstime

Черговий великий європейський фотостік, який продає зображення поштучно і по абонементу. Розцінки при поштучної продажу рознесені на щаблі виходячи з рангу автора зображень. Як і інші фотостоки, Dreamstime дає можливість викачувати безкоштовні фотографії.

Приймає: фотографії розміром більше 3 мегапікселів, також векторні зображення.

Розцінки для клієнтів: від 1 до 30 дол за RF ліцензії, до 300 дол за ERF ліцензії, від 350 дол за права на зображення.

Винагорода авторові: 50% від суми, мінімальна сума для виводу – 100 дол за вимогою

Особливості при реєстрації: немає.

123RF

Фотостік, аналогічний Shutterstok продає зображення за абонементами і поштучно. Причому вартість абонента менше, а скачати за день можна більше.

Приймає: фотознімки розміром більше 4 мегапікселів, а також векторні зображення.

Розцінки для клієнтів: абонент на місяць 89 – 199 дол, поштучно – 1-3 дол за RF-ліцензії, 999 дол за ERF-ліцензії.

Винагорода авторові: 50% (при штучної) або 0,36 дол / штука (за абонементом), мінімальна сума для виводу-100 дол на місяць.

Особливості при реєстрації: 10 тестових робіт.

BigStockPhoto

Менш солідний в порівнянні з попередніми, проте досить таки стабільний фотобанк продає зображення поштучно. Пропонує придбання зображень не

тільки лише по RF і ERF ліцензіями, але також по 10-й спеціальними ліцензіями, які задовольнять будь-якого дизайнера.

Приймає: фотознімки розміром більше 2 мегапікселів.

Розцінки для клієнтів: від 1 дол (RF ліцензія), від 15 до 120 дол (ERF ліцензія).

Винагорода: 50% від продажів, мінімальна сума для виводу – 30 дол.

Особливості при реєстрації: теоретичний іспит.

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 маркування даних мікропейментових фотобанків з застосуванням методів 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

Проаналізуємо методи цифрової (комп'ютерної) стеганографії. У зв'язку з бурхливим розвитком сучасних комп'ютерних технологій, підвищенням обсягів інформації, що передається і обробляється в комп'ютерних системах та мережах, появою нових інформаційних послуг та сервісів актуальність захисту авторських прав на дані мікропейментових фотобанків неухильно зростає. Перспективні стеганосистеми повинні володіти поліпшеними характеристиками, загальні вимоги до них за різноманітними показниками ефективності узагальнено в табл. 3.1.

Проведені дослідження показали, що на сьогоднішній день в області розробки, побудови та експлуатації стеганографічних засобів захисту авторських прав на дані мікропейментових фотобанків об'єктивно існують наступні суперечливі фактори. З одного боку зросли обсяги даних, що обробляються і передаються в комп'ютерних системах та мережах, підвищилися ймовірнісно-часові вимоги до безпеки, вірогідності і пропускної здатності стеганографічних каналів передачі даних, з'явилися нові загрози інформаційній безпеці, в тому числі, пов'язані з афінними перетвореннями. З іншого боку, існуючий на сьогоднішній день математичний апарат стеганографічного захисту авторських прав на дані мікропейментових фотобанків, не дозволяє реалізувати забезпечення цих підвищених вимог. Перераховані фактори в сукупності складають об'єктивно існуюче протиріччя, на рішення якого і спрямована мета даної роботи. Перспективним напрямком у вирішенні виявленого протиріччя є стеганографічні методи та засоби захисту авторських прав на дані мікропейментових фотобанків, засновані на використанні методів теорії дискретних сигналів і технології прямого розширення спектру.

Таблиця 3.1 – Загальні вимоги до перспективних стеганографічних систем захисту авторських прав на дані мікропейментових фотобанків

Вимоги до безпеки (стеганографічної стійкості)	
1.	Здатність протидіяти відомим методам стеганографічного аналізу і мати запас стійкості із врахуванням тенденцій розвитку засобів електронної обчислювальної техніки й стеганографічної науки.
2.	Стеганоперетворення, що застосовуються, повинні базуватися на надійній та прозорій математичній базі і не мати вбудованих «недокументованих можливостей».
Вимоги до вірогідності інформації, що обробляється і передається	
1.	Ймовірність помилкового вилучення інформаційних повідомлень на приймальній стороні не повинна перевершувати наперед заданої величини (була забезпечена безпомилкова передача даних).
2.	Стеганографічне перетворення не повинне вносити додаткових спотворень у дані, які вбудовують/витягаються.
Вимоги за величиною внесених спотворень у контейнери	
1.	Спотворення, що вносяться у контейнери, повинні бути нижче деякого критичної межі, наприклад, рівня межі чутливості людини до спотворення контейнерів-зображень.
2.	Рівень спотворень, що вносяться, не повинен знижувати «експлуатаційні» характеристики контейнерів.
Вимоги до реалізації стеганоалгоритмів	
1.	Швидкодія стеганоалгоритмів має бути не нижче, ніж швидкодія існуючих алгоритмів стеганографічного перетворення.
2.	Операції, які використовуються у стеганоалгоритмах, повинні мати ефективну програмну та апаратну реалізацію.
Вимоги до пропускної здатності стеганоканалів, що організовуються	
1.	Пропускна здатність повинна бути максимізована.
2.	Підвищення пропускної здатності не повинне призводити до зниження безпеки та вірогідності даних, що обробляються і передаються, а також до підвищення внесених спотворень у стеганоконтейнери.

Цей підхід дозволяє забезпечити високі показники безпеки стеганосистем, стеганоканали, що організовуються, володіють всіма перевагами широкосмугових систем зв'язку, а саме високим рівнем безпеки й вірогідністю передачі даних, стійкістю до їх несанкціонованого ознайомлення (вилучення) та детектування.

Досліджемо формальний математичний опис і структурну схему секретної системи. За аналогією з теорією секретних систем К. Шеннона вводяться основні елементи та математичні оператори, які абстрактно описують стеганографічну систему захисту авторських прав на дані мікропейментових фотобанків. У введеній формалізації отримано визначення крихких і робастних стеганосистем, а також вводяться ймовірнісні показники, які характеризують апостеріорні знання зловмисника про секретні ключі і повідомлення, що вбудовуються. Вводиться поняття стеганосистеми, яка теоретично не детектується, обґрунтовується необхідна і достатня умова для її реалізації. Обґрунтовуються критерії та показники ефективності стеганозахисту, математично формалізується постановка науково-прикладної проблеми бакалаврського дослідження.

Розглянемо структуру стеганографічної системи стеганографічної системи та окремі елементи математичної моделі, що пропонується: $I = \{I_0, I_1, \ldots, I_m\}$ – множина інформаційних повідомлень, які формуються джерелом, де символом I_0 формально позначено «порожнє повідомлення, тобто позначення I_0 відповідає випадку, коли джерело інформації не формує жодних повідомлень, які підлягають передачі каналами зв'язку; $M = \{M_0, M_1, \ldots, M_m\}$ – множина інформаційних даних, отриманих на виході прекодера після відповідного перетворення, де символом M_0 формально позначено результат попереднього кодування «порожнього повідомлення»; $L = \{L_1, L_2, \ldots, L_l\}$ – множина порожніх контейнерів, які формуються джерелом контейнерів і призначені для вбудовування даних; $B = \{B_1, B_2, \ldots, B_l\}$ – множина сформованих даних про особливості природної надлишковості порожніх контейнерів, які формуються

блоком урахування особливостей контейнерів (БУОК), і призначених для реалізації вбудовування даних; $E = \{E_1, E_2, \ldots, E_n\}$ – множина можливих стеганограм (заповнених контейнерів), що формуються пристроєм стеганографічного кодування (стеганокодером); $\varepsilon = \{\varepsilon_1, \varepsilon_2, ..., \varepsilon_e\}$ – множина можливих конфігурацій помилок, що впливають на стеганоконтейнери, які передаються каналами передачі даних (ПД); $E^* = \{E_1^*, E_2^*, ..., E_r^*\}$ – множина можливих спотворених під час передачі каналами ПД стеганограм (спотворених заповнених контейнерів); $M^* = \{M_1^*, M_2^*, ..., M_r^*\}$ – множина можливих «оцінок» вбудованих інформаційних даних як результат декодування прийнятої (і, можливо, спотвореної) стеганограми; $S = \{Так, Ні\}$ – множина можливих рішень детектора про приналежність витягненої оцінки множини «ненульових» інформаційних даних; $\varphi = \{\varphi_1, \varphi_2, \ldots, \varphi_k\}$ – множина прямих відображень $\varphi_i: M \to E$, $i = 1, 2, \ldots, k$; $\varphi^{-1} = \{\varphi_1^{-1}, \varphi_2^{-1}, \ldots, \varphi_k^{-1}\}$ – множина зворотних відображень $\varphi^{-1}{}_i: E \to M$, $i = 1, 2, \ldots, k$; $K = \{K_1, K_2, \ldots, K_k\}$ – множина ключів прямого стеганографічного перетворення (стеганокодування), причому кожне відображення $\varphi_i \in \varphi$ задається ключем K_i; $K^* = \{K_1^*, K_2^*, \ldots, K_k^*\}$ – множина ключів зворотного стеганографічного перетворення (стеганодекодування), причому кожне зворотне відображення $\varphi_i^{-1} \in \varphi^{-1}$ задається ключем K^*_i; $\{W_I\}$ – оператор формування інформаційних повідомлень $I_j \in \{I_1, I_2, \ldots, I_m\}$; $\{W_M\}$ – оператор попереднього кодування інформаційних повідомлень, результатом дії якого є інформаційні дані $M_j \in M$; $\{W_L\}$ – оператор формування порожніх контейнерів $L_u \in \{L_1, L_2, \ldots, L_l\}$, призначених для вбудовування інформаційних даних; $\{W_B\}$ – оператор дослідження і оцінювання особливостей контейнерів, що формуються, результатом дії якого є вироблені правила і обмеження B_u на вбудовування інформаційних даних у сформовані контейнери $L_u \in L$; $\{W_E\}$ – оператор прямого стеганографічного перетворення (стеганокодування) інформаційних даних $M_j \in M$ і порожніх контейнерів $L_u \in L$ у стеганограми

(заповнені контейнери) $E_i \in E$; $\{W_K\}$ – оператор формування ключових даних $K_i \in K$ и $K^*_i \in K^*$; $\{W^{-1}_E\}$ – оператор зворотного стеганографічного перетворення (стеганодекодування) стеганограм (заповнених контейнерів) $E_i \in E$ у дані, які витягнуто, $M^*_j \in M^*$, результатом дії оператора $\{W^{-1}_E\}$ є деяка «оцінка» M^*_j вбудованих інформаційних даних M_j; $\{W_S\}$ – оператор детектування отриманих інформаційних даних, дія якого полягає в ототожненні одержаної «оцінки» M^*_j з одним зі елементів множини інформаційних даних M; $\{W^{-1}_M\}$ – оператор зворотного перетворення (декодування) отриманих інформаційних даних $M_j \in M$ у інформаційні повідомлення $I_j \in I$; $\{W^{-1}_I\}$ –оператор обробки отриманих інформаційних повідомлень $I_j \in I$ на приймальній стороні; $\{W^{-1}_{M^*}\}$ – оператор дій противника щодо реалізації безключового детектування, тобто детектування інформаційних даних $M_i \in M$ без знання ключових даних $K^*_i \in K^*$ зворотного стеганоперетворення (стеганодекодування); $\{W^{-1}_{K^*}\}$ – оператор дій противника щодо реалізації пошуку секретного ключа $K^*_i \in K^*$ зворотного стеганоперетворення (стеганодекодування).

Отримане на приймальній стороні інформаційне повідомлення I_j є результатом виконання багатоетапного стеганографічного перетворення, що складається з передобробки, стеганокодування і декодування, детектування та завершальній обробці в декодері інформаційних повідомлення. Формально результат такого багатократного перетворення запишемо у вигляді:

$$I_j = W_M^{-1}\left(\begin{matrix} W_S\left(W_E^{-1}\left(W_{K^*}(K_i^*), W_R\left(W_E\left(W_K(K_i), W_L(L_u), W_B(W_L(L_u)), W_M(W_I(I_j))\right)\right)\right)\right), \\ W_E^{-1}\left(W_{K^*}(K_i^*), W_R\left(W_E\left(W_K(K_i), W_L(L_u), W_B(W_L(L_u)), W_M(W_I(I_j))\right)\right)\right) \end{matrix}\right).$$

Припустимо, що зловмисник може перехопити стеганограму E_w. З її допомогою він може спробувати обчислити апостеріорні ймовірності вбудовування різних можливих інформаційних даних:

$$P_{M|E_w} = \{P(M_0|E_w), P(M_1|E_w), ..., P(M_m|E_w)\}, \quad (3.1)$$

і різних можливих ключів стеганодекодування:

$$P_{K^*|E_w} = \left\{P\left(K_1^*|E_w\right), P\left(K_2^*|E_w\right), ..., P\left(K_k^*|E_w\right)\right\}, \quad (3.2)$$

які можуть бути використані під час зворотного стеганоперетворення.

За аналогією з теорією секретних систем назвемо задачу обчислення апостеріорних імовірностей (3.1) *завданням безключового детектування*, а завдання обчислення апостеріорних імовірностей (3.2) *завданням пошуку секретного ключа стеганодекодування*. Множини апостеріорних імовірностей (3.1) і (3.2) утворюють апостеріорні знання зловмисника про ключі $K^* = \{K_1^*, K_2^*, \ldots, K_k^*\}$ і інформаційні дані $M = \{M_0, M_1, \ldots, M_m\}$ після перехоплення стеганограми E_w. Принциповою відмінністю стеганосистеми від секретної (крипто-) системи є наявність у множині M «нульових» інформаційних даних M_0, а обчислення апостеріорних ймовірностей (3.1) зловмисником саме й виконується з метою правильного детектування цих «нульових» або «ненульових» повідомлень. Тобто основним завданням зловмисника є імітація роботи детектора без знання секретного ключа стеганодекодування, а множини $P_{M|E_w}$ та $P_{K^*|E_w}$ є множинами висунутих їм припущень, яким приписані відповідні апостеріорні ймовірності.

За допомогою розглянутого теоретико-інформаційного підходу до формального опису стеганографічних систем, за аналогією з теорією секретних систем К. Шеннона, введемо визначення *цілковитої стійкості (теоретичної недетектованості) стеганосистеми*: якщо для всіх стеганограм з множини $E = \{E_1, E_2, \ldots, E_n\}$ апостеріорні ймовірності (3.1), (3.2) дорівнюють відповідним апріорним ймовірностям $P_M = \left\{P\left(M_0\right), P\left(M_1\right), ..., P\left(M_m\right)\right\}$ та $P_{K^*} = \left\{P\left(K_1^*\right), P\left(K_2^*\right), ..., P\left(K_k^*\right)\right\}$ незалежно від їх величини, тоді перехоплення стеганограми E_w не дає зловмисникові жодної додаткової інформації щодо вбудованих інформаційних даних.

Стеганографічні системи, які задовольняють цьому визначенню, будемо називати *цілком стійкими (теоретично недетектованими) стеганосистемами*. Якщо умова теоретичної недетектованості є не виконаною, тоді для деяких інформаційних даних їх апостеріорні ймовірності відрізняються від апріорних, що дає зловмиснику деяку додаткову інформацію для стегоаналізу, тобто для вирішення задачі безключового детектування.

Розглянемо задачу безключового детектування, тобто задачу обчислення зловмисником значень апостеріорних ймовірностей (3.1) ототожнення отриманої оцінки $M*_j$ елементу множини різних можливих інформаційних даних. Використовуючи формулу Байєса для довільного елемента множини $P_{M|E_w} = \{P(M_0|E_w), P(M_1|E_w), ..., P(M_m|E_w)\}$ запишемо:

$$P(M_j|E_w) = \frac{P(M_j)P(E_w|M_j)}{P(E_w)}, \tag{3.3}$$

де $P(E_w|M_j)$ – умовна ймовірність формування стеганограми E_w за умови того, що вбудовуванню піддавались інформаційні дані M_j; $P(E_w)$ – ймовірність формування криптограми E_w.

Умовна ймовірність $P(E_w|M_j)$ дорівнює сумі ймовірностей таких секретних ключів, які призводять до формування стеганограми E_w в результаті вбудовування інформаційних даних M_j:

$$P(E_w|M_j) = \sum_i \left(P(K_i)|E_w = \varphi_i(K_i, M_j)\right).$$

Використовуючи (3.3) необхідну і достатню умову теоретичної недетектованості (умову досконалої стійкості стеганосистеми) перепишемо у вигляді рівності: $P(E_w|M_j) = P(E_w)$, тобто сума ймовірностей секретних ключів, що призводять до формування стеганограми E_w в результаті вбудовування

інформаційних даних повідомлення M_j, не повинна залежати від вибору цього повідомлення:

$$\forall j,l \in [1,2,...,m]: P(E_w|M_j) = P(E_w|M_l) = P(E_w). \quad (3.4)$$

Фактично це означає, що в теоретично недетектованій стеганосистемі зловмисник може тільки вгадати, вбудовані «ненульові» інформаційні дані в перехоплений контейнер чи ні. Тобто для нього апостеріорні ймовірності «нульових» і «ненульових» повідомлень дорівнюють апріорним.

Враховуючи функціональне призначення стеганосистеми, введемо наступні показники ефективності: пропускна здатність стеганоканала, який організовується, – Q; обсяг ключових даних – l_K; стійкість стеганосистеми (ймовірність P^*_K знаходження зловмисником ключа детектування (вилучення) повідомлень, ймовірність P^*_M несанкціонованого детектування (вилучення) повідомлення без знання секретного ключа (ймовірність безключового детектування (читання) повідомлень), безпечний час $T_Б$ роботи стеганосистеми); величина спотворень, які вносяться, I^* у контейнер, що використовується; ймовірність $P_{пом.вил.}$ помилкового вилучення інформаційних даних повідомлення.

Множина введених показників і критеріїв їх оцінювання дозволяє ввести узагальнений показник ефективності стеганозахисту як функціонал:

$$W = F(Q, l_K, P_K, P_M, T_Б, I^*, P_{пом.вил.}), \quad (3.5)$$

де вигляд $F(Q, l_K, P_K, P_M, T_Б, I^*, P_{пом.вил.})$ та конкретний внесок окремих (часткових) показників $Q, l_K, P_K, P_M, T_Б, I^*, P_{пом.вил.}$ визначається конкретним призначенням системи, особливостями її використання та умовами експлуатації.

До безумовних відносяться показники безпеки (стеганостійкості) l_K, P_K, P_M, $T_Б$, втрати вірогідності $P_{пом.вил.}$та величини внесених спотворень I^*. Їх оцінювання проводиться за допомогою булевих змінних w_i, $i-1,..,6$, які приймають значення «1» (істина), якщо відповідна вимога виконується і «0»

(неправда) – в іншому випадку. Функція відповідності $f_в(w_1, w_2, ..., w_6)$ дає оцінку того, чи задовольняє розглянута стеганосистема безумовним критеріям. Як умовний будемо використовувати показник пропускної здатності Q. Таким чином, постановку проблеми дослідження сформулюємо як однокритеріальну оптимізаційну задачу:

$$\left\langle \max(Q): l_K^{T1} \le l_K \le l_K^{T2}, P_K \le P_K^T, P_M \le P_M^T, T_Б \ge T_Б^T, P_{пом.вил.} \le P_{пом.вил.}^T, I^* \le I^{*T} \right\rangle,$$

або, що еквівалентно, $\left\langle \max(Q): f_в(w_1, w_2, ..., w_6) = 1 \right\rangle$.

Проаналізуємо модель передачі даних у системах зв'язку з прямим розширенням спектру. На основі проведеного аналізу пропонується математична модель і структурна схема стеганозахисту авторських прав на дані мікропейментових фотобанків з використанням технології прямого розширення спектру. Розробляються методи та обчислювальні алгоритми вбудовування й вилучення інформаційних повідомлень.

Для експериментального дослідження ефективності вбудовування повідомлень в нерухомі зображення з використанням складних дискретних сигналів і технології прямого розширення спектру розроблено імітаційну комп'ютерну модель, отримано емпіричні залежності величини внесених спотворень I від пропускної здатності Q, а також залежність I від частоти помилок вилучення $P_{пом.вил.}^* \approx P_{пом.вил.}$

З наведених залежностей випливає, що підвищення пропускної здатності веде до різкого збільшення спотворень I, що вносяться в контейнер-зображення. Непомітні для стороннього спостерігача спотворення (лежачі нижче порогу чутливості зорової системи людини) вносяться лише за умови $Q \le 0.005$. Для фіксованої $Q = 0.005$ отримана емпірична крива, що характеризує залежність I та $P_{пом.вил.}^* \approx P_{пом.вил.}$. Очевидно, що домогтися низьких спотворень, які лежать нижче межі зорової чутливості людини ($I \le 2...3\%$), можна тільки при дуже

високій ймовірності помилкового вилучення інформаційних даних ($P_{пом.вил.} \geq 0.1$).

Таким чином, практичне використання розглянутих стеганосистем сполучено з пошуком компромісу між величиною внесених спотворень I, ймовірністю правильно вилучення повідомлення $1 - P_{пом.вил.}$ на приймальній стороні і пропускною здатністю, що забезпечується, Q. Перспективним слід вважати використання великих ансамблів слабокорельованих дискретних сигналів. Це дозволить, з одного боку, без значного підвищення внесених спотворень в контейнер-зображення істотно підвищити пропускну здатність стеганоканалу, а з іншого – завдяки адаптивному формуванню (вибору) дискретних сигналів істотно знизити ймовірність помилкового вилучення вбудованих даних.

Проаналізуємо відомі методи синтезу дискретних сигналів, в тому числі методи формування послідовностей з особливими кореляційними властивостями. Проведемо порівняльні дослідження властивостей дискретних сигналів, що формуються, вказуються їхні переваги й недоліки. Розвивається окремий напрямок у розвитку методів формування дискретних сигналів, заснований на використанні алгебраїчних і структурних властивостей циклічних орбіт групових кодів та дозволяє синтезувати нові класи дискретних сигналів з багаторівневою функцією кореляції. Показано, що запропонований підхід дозволяє формувати множини дискретних послідовностей з поліпшеними властивостями.

Розглянемо структуру групового (n, k, d) коду V над $GF(q)$ з погляду циклічних властивостей, які утворюють його послідовності. Будемо використовувати при цьому поняття *циклічної орбіти* V_ξ – множину послідовностей з елементами із $GF(q)$, еквівалентних один одному щодо операції циклічного зсуву, тобто множину таких:

$$C_i = \left(c_0^i, c_1^i, \ldots c_{n-1}^i\right),\ c_v^i \in GF(q) \text{ та } C_j = \left(c_0^j, c_1^j, \ldots c_{n-1}^j\right),\ c_v^j \in GF(q),$$

що виконується рівність:

$$\left(c_0^i, c_1^i, \ldots c_{n-1}^i\right) = \left(c_{(\tau) \bmod (n)}^j, c_{(\tau+1) \bmod (n)}^j, \ldots c_{(\tau+n-1) \bmod (n)}^j\right), \quad (3.6)$$

для будь-якого $\tau \in \{0, \ldots, n-1\}$.

Розглянемо множину W всіх n-послідовностей з елементами з $GF(q)$, які утворюють так званий «повний код». Структура множини еквівалентна векторному простору $GF^n(q)$ з покомпонентним додаванням та множенням на скаляр. Розіб'ємо всю множину W на підмножини орбіт V_0, V_1,…, V_L , кожна з яких містить сукупність послідовностей, еквівалентних один одному відносно операції циклічного зсуву (3.6). Таким чином, отримаємо розкладання векторного простору $GF^n(q)$ на множини непересічних орбіт. Символами $C_{i,j} \in GF^n(q)$, $i = 0, \ldots, L$, $j = 1, \ldots, Z_i$ схематично позначено n-послідовності, як елементи множини W. Всі $C_{i,j}$ згруповані за ознакою еквівалентності відносно операції циклічного зсуву. Кожна група – суть множини V_i, всі елементи множини V_i утворюють i-ту орбіту множини W.

Груповий код однозначно задається лідерами (представниками) складових його циклічних орбіт. Дистанційні (кореляційні) властивості лідерів орбіт визначаються дистанційними властивостями групових кодів, при цьому еквівалентність відносно операції циклічного зсуву відсутня за визначенням. Цю властивість покладемо в основу формування ансамблю дискретних сигналів: на основі перетину ненульових циклічних орбіт (вибору лідера кожної орбіти) сформуємо безліч дискретних послідовностей з дистанційними властивостями вихідного групового коду і не еквівалентних один одному відносно операції циклічного зсуву.

Покажемо розкладання векторного простору $GF^n(q)$ на множини непересічних орбіт V_ξ, $\xi = 0, \ldots, L$, представлення групового коду V через об'єднання кінцевої кількості орбіт та схема вибору лідерів орбіт – по одному довільному представникові з кожної циклічної підмножини V_ξ, $\xi = 0, \ldots, M$ (для

зручності позначення кодові слова $C_{v,u}=\left(c_0^{v,u},c_1^{v,u},...c_{n-1}^{v,u}\right)$ позначені двома індексами: v – номер орбіти V_v коду V, $v=1,...,M$; u – номер кодового слова в орбіті, $u=1,...,z_v$, де z_v – кількість кодових слів в орбіті V_v, $z_v \le n-1$). Із відібраних представників орбіт сформуємо множину $S=\left(S_1,S_2,...,S_M\right)$, де $S_v=C_{v,u}$, $v=1,...,M$, а вибір індексу u при відповідному $C_{v,u}$ визначається правилом перетину v-ї циклічною орбіти групового коду. Розглянемо двійковий випадок, тобто обмежимося дослідженням властивостей множини $S=\left(S_1,S_2,...,S_M\right)$, утвореної шляхом перетину циклічних орбіт двійкового групового коду. Елементи дискретних послідовностей, що формуються, (дискретних сигналів) $S_v=\left(s_0^v,s_1^v,...s_{n-1}^v\right)$ задамо за елементами відібраних кодових слів (лідерів орбіт) $C_{v,u}=\left(c_0^{v,u},c_1^{v,u},...c_{n-1}^{v,u}\right)$ наступним чином: $s_i^v=\begin{cases}1, c_i^{v,u}=1;\\ -1, c_i^{v,u}=0.\end{cases}$.

Припустимо, що розглянутий код V має ваговий спектр виду:

$$A(w)=\begin{cases}1, w=0;\\ 0, 1\le w\le d-1;\\ \ne 0, w\ge d,\end{cases}$$

$w=0,...,n$, $A(w)$ – кількість кодових слів коду V з вагою w.

Тоді утворена перетином циклічних орбіт коду V множина двійкових сигналів $S=\left(S_1,S_2,...,S_M\right)$ володіє наступними властивостями.

Твердження.

1. Бокові пелюстки періодичної функції авто– (ПФАК) і взаємної (ПФВК) кореляції приймають наступні значення:

$$ПФВК, ПФАК=\frac{n-2w}{n},\ \text{для таких}\ w=d,d+1,...,n,\ \text{що}\ A(w)\ne 0. \quad (3.7)$$

2. Для всіх $w=d,d+1,...,n$, таких, що $A(w)=0$ бокові пелюстки ПФАК та ПФВК ніколи не приймають значень $\frac{n-2w}{n}$.

3. Потужність M ансамблю $S=(S_1,S_2,...,S_M)$ визначається кількістю ненульових орбіт коду V та обмежена знизу виразом: $M \geq \frac{2^k-1}{n}$.

Таким чином, для синтезу дискретних сигналів з необхідними ансамблевими і кореляційними властивостями, слід шукати компроміс між потужністю коду (потужністю відповідного ансамблю сигналів) і його дистанційними властивостями (кореляційними властивостями дискретних сигналів).

Проведені дослідження кореляційних та ансамблевих властивостей дискретних сигналів показали, що використання запропонованого методу дозволяє синтезувати нові класи дискретних сигналів з багаторівневими функціями кореляції та поліпшеними ансамблевими властивостями. Перехід до наступного класу дискретних сигналів сполучений із зміною кількості співмножників у перевірочному многочлені групового коду, що використовується.

Розглянемо випадок, коли двійковий груповий код (n,k,d) задано через перевірочний многочлен виду:

$$h(x)=f_{i1}(x)f_{i_2}(x)f_{i_3}(x)=\prod_{s=1}^{m-1}\left(x-\alpha^{i_1(2^s)}\right)\left(x-\alpha^{i_2(2^s)}\right)\left(x-\alpha^{i_3(2^s)}\right),$$

де $f_{i_1}(x)$, $f_{i_2}(x)$ та $f_{i_3}(x)$ – мінімальні многочлени елементів $\alpha^{i_1}\in GF(2^m)$, $\alpha^{i_2}\in GF(2^m)$ і $\alpha^{i_3}\in GF(2^m)$ відповідно, де порядок елементів α^{i_1}, α^{i_2} і α^{i_3} рівний порядку мультиплікативної групи кінцевого поля $GF(2^m)$, $n=2^m-1$, α – примітивний елемент кінцевого поля $GF(2^m)$, $n=2^m-1$.

Оцінимо ваговій спектр коду, кореляційні та ансамблеві властивості дискретних сигналів, що формуються:

$$A(w)=\begin{cases}1, w=0;\\ 0, w=1,...,2^{m-1}-2^{\frac{m+1}{2}}-1;\\ \neq 0, w=2^{m-1}-2^{\frac{m+1}{2}};\\ 0, w=2^{m-1}-2^{\frac{m+1}{2}}+1,...,2^{m-1}-2^{\frac{m+1}{2}-1}-1;\\ \neq 0, w=2^{m-1}-2^{\frac{m+1}{2}-1};\\ 0, w=2^{m-1}-2^{\frac{m+1}{2}-1}+1,...,2^{m-1}-1\\ \neq 0, w=2^{m-1};\\ 0, w=2^{m-1}+1,...,2^{m-1}+2^{\frac{m+1}{2}-1}-1;\\ \neq 0, w=2^{m-1}+2^{\frac{m+1}{2}-1};\\ 0, w=2^{m-1}+2^{\frac{m+1}{2}-1}+1,...,2^{m-1}+2^{\frac{m+1}{2}}-1;\\ \neq 0, w=2^{m-1}+2^{\frac{m+1}{2}};\\ 0, w=2^{m-1}+2^{\frac{m+1}{2}}+1,...,2^{m}-1,\end{cases}$$

$$ПФВК, ПФАК=\begin{cases}\frac{-1-2^{\frac{m+1}{2}+1}}{2^m-1}, w=2^{m-1}-2^{\frac{m+1}{2}};\\ \frac{-1-2^{\frac{m+1}{2}}}{2^m-1}, w=2^{m-1}-2^{\frac{m+1}{2}-1};\\ \frac{-1}{2^m-1}, w=2^{m-1};\\ \frac{-1+2^{\frac{m+1}{2}}}{2^m-1}, w=2^{m-1}+2^{\frac{m+1}{2}-1};\\ \frac{-1+2^{\frac{m+1}{2}+1}}{2^m-1}, w=2^{m-1}+2^{\frac{m+1}{2}},\end{cases}$$

$$M=\frac{2^{3m}-1}{2^m-1}=2^{2m}+2^m+1.$$

Узагальнимо отримані вище результати. Для цього розглянемо двійковий груповий (n,k,d) код над $GF(2)$, заданий через перевірочний многочлен:

$$h(x)=f_{i1}(x)f_{i_2}(x)...f_{i_u}(x)=\prod_{s=1}^{m-1}\left(x-\alpha^{i_1\left(2^s\right)}\right)\left(x-\alpha^{i_2\left(2^s\right)}\right)...\left(x-\alpha^{i_u\left(2^s\right)}\right), \quad (3.8)$$

де $f_{i_1}(x),\ f_{i_2}(x)\ \ldots\ f_{i_u}(x)$ – мінімальні многочлени елементів $GF\left(2^m\right)$.

Перевірочний многочлен коду з параметрами (3.8) містить як співмножники перевірочні многочлени всіх кодів, з перевірочними многочленами $h(x)=f_{i1}(x)f_{i_2}(x)...f_{i_y}(x), y\le u$. Звідси випливає, що ненульові компоненти вагового спектра утворюються почерговим додаванням (в порядку додавання співмножників у многочлен $h(x)$) відповідної пари елементів (для всіх $y=2,3,...,u$): $A(z_y)\neq 0$, $A(2^m-z_y)\neq 0$, де:

$$z_y = \max_{s=0,\dots,m-1}\left\{\left(2^s\right)\bmod\left(2^m-1\right),\left(i_2 2^s\right)\bmod\left(2^m-1\right),\dots,,\left(i_y 2^s\right)\bmod\left(2^m-1\right)\right\}.$$

Звідки отримаємо оцінки:

$$ПФВК, ПФАК = \begin{cases} \dfrac{2^m-2z_u-1}{2^m-1}, w=z_u= \\ =\max\limits_{s=0,\dots,m-1}\left\{\left(2^s\right)\bmod\left(2^m-1\right),\left(i_2 2^s\right)\bmod\left(2^m-1\right),\dots,,\left(i_u 2^s\right)\bmod\left(2^m-1\right)\right\}; \\ \dots \\ \dfrac{2^m-2z_3-1}{2^m-1}=\dfrac{-1-2^{\frac{m+1}{2}+1}}{2^m-1}, w=z_3=2^{m-1}-2^{\frac{m+1}{2}}; \\ \dfrac{2^m-2z_2-1}{2^m-1}=\dfrac{-1-2^{\frac{m+1}{2}}}{2^m-1}, w=z_2=2^{m-1}-2^{\frac{m+1}{2}-1}; \\ \dfrac{2^m-2z_1-1}{2^m-1}=\dfrac{-1}{2^m-1}, w=z_1=2^{m-1}; \\ \dfrac{2^m-2(2^m-z_2)-1}{2^m-1}=\dfrac{-1+2^{\frac{m+1}{2}}}{2^m-1}, w=2^m-z_2=2^{m-1}+2^{\frac{m+1}{2}-1}; \\ \dfrac{2^m-2(2^m-z_3)-1}{2^m-1}=\dfrac{-1+2^{\frac{m+1}{2}+1}}{2^m-1}, w=2^m-z_3=2^{m-1}+2^{\frac{m+1}{2}}; \\ \dots \\ \dfrac{2^m-2(2^m-z_u)-1}{2^m-1}, w=2^m-z_u= \\ =2^m-\max\limits_{s=0,\dots,m-1}\left\{\left(2^s\right)\bmod\left(2^m-1\right),\left(i_2 2^s\right)\bmod\left(2^m-1\right),\dots,,\left(i_u 2^s\right)\bmod\left(2^m-1\right)\right\}, \end{cases}$$

$$M=\frac{2^{um}-1}{2^m-1}=2^{(u-1)m}+2^{(u-2)m}+\dots+2^m+1.$$

Таким чином, додавання мінімального многочлена як чергового співмножника у (3.8) підвищує потужність ансамблю на $2^{(u-i)m}$, де $u-i$ – кількість доданих мінімальних многочленів.

3.2 Розробка структурної схеми

Відомий метод стеганографічного приховування даних в просторовій області зображень із використанням прямого розширення спектру [1-3] ґрунтується на тому, що на передавальній стороні після шифрування та перешкодостійкого кодування окремі блоки $m_i = (m_{i_0}, m_{i_1}, ..., m_{i_{k-1}}),\ i = 0,..., N-1$ даних інформаційного повідомлення $m = (m_0, m_1, ..., m_{N-1})$ за допомогою відповідних пристроїв модулюються шумоподібними дискретними сигналами:

$$\Phi_i = (\varphi_{i_0}, \varphi_{i_1}, ..., \varphi_{i_{n-1}}),\ \Phi_i \in \Phi = \{\Phi_0, \Phi_1, ..., \Phi_{M-1}\},\ k \le M,$$

із базою $B = TF$, де T – тривалість елемента сигналу φ_{i_j}, F – полоса частот сигналу Φ_i. Оскільки $F = n\frac{1}{T}$ маємо $B = n >> 1$ і база сигналу задає кратність розширення полоси частот сигналу Φ_i по відношенню до елементарних сигналів φ_{i_j} та/або m_{i_j}.

В результаті для кожного інформаційного блоку m_i формується блок модульованого інформаційного сигналу:

$$E_i = \sum_{j=0}^{k-1} m^*_{i_j} \Phi_j = \left(\sum_{j=0}^{k-1} m^*_{i_j} \varphi_{j_0}, \sum_{j=0}^{k-1} m^*_{i_j} \varphi_{j_1}, ..., \sum_{j=0}^{k-1} m^*_{i_j} \varphi_{j_{n-1}} \right), \qquad (3.9)$$

де:

$$m^*_{i_j} = \begin{cases} +1, m_{i_j} = 1; \\ -1, m_{i_j} = 0; \end{cases}$$

який за статистичними властивостями приймає вигляд випадкової послідовності, а за рахунок великої бази дискретних сигналів досягається розширення спектру частот в $B = n$ разів.

Отримане модульоване повідомлення E_i подається на пристрій перемежування, на якому елементи E_i за допомогою таємного ключа K_1 перемішуються за відповідним правилом f. Отримані данні $\overline{E_i} = f(E_i, K_1)$ за

допомогою відповідного пристрою поелементо додаються до даних контейнера C_i (даних цифрового зображення в просторовій області) за правилом:

$$S_i = C_i + \overline{E_i} \cdot G,$$

де $G > 0$ – коефіцієнт підсилення розширювального сигналу, який задає «енергію» вбудованих блоків інформаційного повідомлення.

Отримані дані S_i подаються на пристрій квантування, який виконує певне перетворення для зберігання початкового динамічного діапазону зображення-контейнеру, в результаті чого формуються окремі блоки стеганограми $\overline{S_i}$ та заповнений контейнер $\overline{S} = \overline{S_0} \cup \overline{S_1} \cup ... \cup \overline{S_{N-1}}$, який передається приймальній стороні.

На приймальній стороні отримані блоки стеганограми $\overline{S_i}$ після фільтрації подаються на пристрій зворотного перемежування, на якому елементи відфільтрованих блоків стегенограми $\overline{\overline{S_i}}$ за допомогою таємного ключа перемішуються за правилом f^{-1}, яке інверсне правилу перемежування f на передавальній стороні. Вилучення блоків інформаційних даних виконується за допомогою кореляційного приймача, який обраховує значення коефіцієнту кореляції отриманих після зворотного перемежування даних $S^*_i = f^{-1}(\overline{\overline{S_i}}, K_1)$ та відповідних дискретних сигналів Φ_j, тотожних тим, що застосовувалися на передавальній стороні:

$$\rho\left(S^*_i, \Phi_j\right) = \frac{1}{n}\sum_{z=0}^{n-1} S^*_{i_z} \varphi_{j_z} \approx G \cdot \frac{1}{n}\sum_{z=0}^{n-1} E_{i_z} \varphi_{j_z} + \frac{1}{n}\sum_{z=0}^{n-1} C_{i_z} \varphi_{j_z}. \qquad (3.10)$$

Припустимо, що масив даних блоку контейнера C_i має випадкову статистичну структуру, тобто покладемо, що другий доданок в правій частині виразу (3.10) близький до нуля і їм можна знехтувати. Тоді маємо:

$$\rho\left(S^*_i,\Phi_j\right)\approx G\cdot\frac{1}{n}\sum_{z=0}^{n-1}E_{i_z}\varphi_{j_z}=G\cdot\frac{1}{n}\sum_{z=0}^{n-1}\left(\sum_{u=0}^{k-1}m^*_{i_u}\,\varphi_{u_z}\right)\varphi_{j_z}=$$
$$=G\cdot\sum_{u=0}^{k-1}m^*_{i_u}\sum_{z=0}^{n-1}\varphi_{u_z}\varphi_{j_z}=G\cdot\sum_{u=0}^{k-1}m^*_{i_u}\,\rho\left(\Phi_u,\Phi_j\right). \quad (3.11)$$

Оскільки всі послідовності із множини Ф формуються за допомогою генератора псевдовипадкових послідовностей, ініційованого таємним ключем K_2, відповідні дискретні сигнали є слабокорельованими, тобто при $u\neq j$ маємо $\rho\left(\Phi_u,\Phi_j\right)\approx 0$. Відповідно до цього всіма доданками, окрім випадку $u=j$, в правій частині рівняння (3) можна знехтувати. Звідки маємо:

$$\rho\left(S^*_i,\Phi_j\right)\approx G\cdot m^*_{i_j}\cdot\frac{1}{n}\sum_{z=0}^{n-1}\left(\varphi_{j_z}\right)^2=G\cdot m^*_{i_j}=\begin{cases}+G;\\-G.\end{cases} \quad (3.12)$$

Відповідне значення вилучених даних приймається за допомогою порогового пристрою відповідно до обрахованого коефіцієнта кореляції. Оскільки $G>0$ і $n>0$ знак $\rho\left(S^*_i,\Phi_j\right)$ в (3.12) залежить тільки від $m^*_{i_j}$, звідки маємо:

$$m^*_{i_j}=sign\left(\rho\left(S^*_i,\Phi_j\right)\right)=\begin{cases}-1,\ \rho\left(S_i,\Phi_j\right)<0;\\+1,\ \rho\left(S_i,\Phi_j\right)>0.\end{cases} \quad (3.13)$$

Якщо $\rho\left(S^*_i,\Phi_j\right)=0$ в (5) будемо вважати, що вбудована інформація була втрачена (стерта).

З вилучених даних на приймальній стороні формуються окремі блоки даних $m_i=\left(m_{i_0},m_{i_1},...,m_{i_{k-1}}\right)$, $i=0,...,N-1$ інформаційного повідомлення $m=\left(m_0,m_1,...,m_{N-1}\right)$, де

$$m_{i_j}=\begin{cases}1,\ m^*_{i_j}=+1;\\0,\ m^*_{i_j}=-1;\end{cases}$$

з яких після перешкодостійкого декодування та розшифрування вилучених даних формуються інформаційні повідомлення. Секретний ключ K_2 задає правило формування псевдовипадкових послідовностей

$\Phi_i = (\varphi_{i_0}, \varphi_{i_1}, ..., \varphi_{i_{n-1}})$, які формуються відповідним генератором та використовуються у якості шумоподібних дискретних сигналів $\Phi_i \in \Phi = \{\Phi_0, \Phi_1, ..., \Phi_{M-1}\}$ з ансамблю Φ потужності M. Правило шифрування та розшифрування на передавальній та приймальній стороні ініціюється секретним ключем K_3.

Застосування пристроїв шифрування та перемежування у процесі приховування та вилучення даних дозволяє покращити статистичні властивості модульованого повідомлення E_i, тобто наблизити його вигляд до випадкової послідовності. Застосування пристроїв перешкодостійкого кодування дозволяє підвищити достовірність передачі інформаційних повідомлень $m = (m_0, m_1, ..., m_{N-1})$ під час стеганографічних перетворень.

Недоліком відомого способу є те, що в процесі вбудовування даних інформаційного повідомлення не враховуються статистичні властивості блоків контейнера C_i, тобто цифрові данні окремих фрагментів просторової області зображення можуть бути корельованими із застосовуваними дискретними сигналами, що призведе до виникнення помилки при вилученні відповідних блоків інформаційних даних на приймальній стороні. Так, наприклад, якщо коефіцієнт кореляції i-го блоку C_i контейнера буде вищий за модулем та протилежний за знаком значенню $G \cdot m^*_{i_j}$, тобто, коли другий доданок в правій частині виразу (3.10) буде перевищувати за модулем та протилежним за знаком першому доданку (та виконуватиметься умова взаємної ортогональності застосовуваних дискретних сигналів), гарантовано відбудеться помилка при вилученні даних за правилом (3.13). На притиці, як довели проведені авторами дослідження, такі випадки відбуваються дуже часто. Це пов'язане з тим, що цифрові данні просторової області реальних зображень, використовуваних під час стеганографічного приховування інформаційних повідомлень, не володіють випадковою статистичною структурою, тобто застосовуване припущення при переході від формули (3.10) до формули (3.11) на практиці не виконується і є

хибним. Зазвичай при стеганографічному приховуванні застосовуються реалістичні зображення і відповідні цифрові данні у просторовій області зображень не є реалізацією випадкового процесу і навіть за своїми статистичними властивостями не подібні до псевдовипадкових послідовностей. Відповідні значення коефіцієнту кореляції:

$$\rho\left(C_i,\Phi_j\right)=\frac{1}{n}\sum_{z=0}^{n-1}C_{i_z}\varphi_{j_z}\neq 0$$

і можуть приймати великі за амплітудою ($\left|\rho\left(C_i,\Phi_j\right)\right|>>1$) та випадкові за знаком величини. Збільшити у цьому випадку достовірність вилучених даних можливо тільки застосувавши низькошвидкісні перешкодостійкі коди, що призводить до зниження відносної швидкості передачі інформації, або підвищивши коефіцієнт підсилення G, що призводить до збільшення внесених похибок.

В основу пропонованого методу поставлена задача створити спосіб стеганографічного в просторовій області зображень із використанням прямого розширення спектру. приховування даних в просторовій області зображень із використанням прямого розширення спектру який, за рахунок врахування статистичних властивостей контейнера C_i дозволить значно підвищити достовірність вилучення вбудованих даних, тобто шляхом введення додаткових обмежень на значення коефіцієнту кореляції використовуваних дискретних сигналів та окремих фрагментів просторової області зображення реалізація корисної моделі дозволить значно зменшити кількість виникаючих помилок при вилученні відповідних блоків інформаційних даних на приймальній стороні.

Поставлена задача вирішується за рахунок адаптивного формування псевдовипадкових послідовностей $\Phi_j=\left(\varphi_{j_0},\varphi_{j_1},...,\varphi_{j_{n-1}}\right)$, із врахуванням статистичних властивостей даних блоків контейнера C_i, тобто значення коефіцієнту кореляції $\rho\left(C_i,\Phi_j\right)$ для всіх $i=0,..,N-1$ та для всіх $j=0,..,M-1$ за модулем не повинно перевищувати деякого наперед визначеного значення $\rho_{\max}$ (значення встановленого порогу):

$$\left|\rho\left(C_i,\Phi_j\right)\right|=\left|\frac{1}{n}\sum_{z=0}^{n-1}C_{i_z}\varphi_{j_z}\right|\le\rho_{\max}. \qquad (3.14)$$

Таким чином, формування послідовностей $\Phi_j\in\Phi=\{\Phi_0,\Phi_1,...,\Phi_{M-1}\}$ виконується за псевдовипадковим правилом, яке ініційоване секретним ключем K_2, та із врахуванням накладеної системи обмежень (6) для всіх $i=0,..,N-1$ та для всіх $j=0,..,M-1$.

При такому формуванні дискретних сигналів кожна послідовність з ансамблю $\Phi=\{\Phi_0,\Phi_1,...,\Phi_{M-1}\}$ не буде корельовано (до встановленої межі) з жодним блоком контейнеру, і відповідно, коефіцієнт кореляції i-го блоку C_i контейнера за модулем ніколи не буде вищий за модулем та протилежним за знаком значенню $\rho_{\max}$. Відповідно до цього (та при виконанні умови взаємної ортогональності застосовуваних дискретних сигналів) другий доданок в правій частині виразу (3.10) може перевищити за модулем та бути протилежним за знаком першому доданку тільки у випадку, коли $\left|G\cdot m^*_{i_j}\right|<\rho_{\max}$. Саме у цьому випадку відбудеться помилка вилучення інформаційних даних, але ймовірність такої події буде значно менша за ймовірність випадку виникнення помилки вилучення даних у відомому способі [4]. Якщо значення порогу $\rho_{\max}$ задати менше, ніж значення коефіцієнту підсилення G, тобто, у випадку, коли виконується нерівність $\left|G\cdot m^*_{i_j}\right|>\rho_{\max}$ помилка не відбудеться зовсім, тобто буде досягнута безпомилкова передача прихованої інформації.

Таким чином досягається конкретний технічний результат, а саме: за рахунок врахування статистичних властивостей цифрових даних окремих фрагментів просторової області контейнера-зображення при формуванні дискретних сигналів вдається значно зменшити кількість виникнення помилок при вилученні відповідних блоків інформаційних даних на приймальній стороні.

Запропонований спосіб стеганографічного приховування та вилучення даних може бути реалізований за допомогою введення відповідного блоку –

пристрою відбору послідовностей за правилом (3.14). Структурна схема системи у вигляді сукупності схеми пристроїв стеганографічного приховування та вилучення даних в просторовій області зображень із використанням прямого розширення спектру, які побудовані за пропонованим способом зображено на рис. 3.1.

Пристрій стеганографічного приховування даних (див. рис. 3.1) працює наступним чином. Джерело інформаційних повідомлень формує послідовність інформаційних даних, які подаються пристрій шифрування, ініційований таємним ключем K_3, що формується джерелом ключів 3. Зашифровані інформаційні повідомлення подаються на пристрій перешкодостійкого кодування, в якому виконується внесення спеціально формованої надмірності для підвищення достовірності інформаційних зашифрованих даних. Джерело ключів 2 формує таємний ключ K_2, який ініціює генератор псевдовипадкових послідовностей. Результатом роботи генератору псевдовипадкових послідовностей є дискретні сигнали, тобто дискретні послідовності, елементи яких сформовано псевдовипадковим чином. Сформовані псевдовипадкові послідовності Φ_j поступають на додатково (в порівнянні із відомим способом) введений пристрій відбору послідовностей за правилом (3.14), в якому для всіх $i = 0,..,N-1$ розраховується значення коефіцієнту кореляції

$$\rho\left(C_i,\Phi_j\right)=\frac{1}{n}\sum_{z=0}^{n-1}C_{i_z}\varphi_{j_z}$$

та порівнюється із наперед визначеним значенням $\rho_{\max}$.

У випадку, коли хоча б для одного $i\in\{0,..,N-1\}$ розраховане значення $\rho\left(C_i,\Phi_j\right)$ перевищить значення порогу $\rho_{\max}$ сформована псевдовипадкова послідовність бракується, тобто дискретні сигнали Φ_j із $\rho\left(C_i,\Phi_j\right)>\rho_{\max}$ хоча б для одного $i\in\{0,..,N-1\}$ для стеганографічного приховування інформаційних даних не застосовуються.

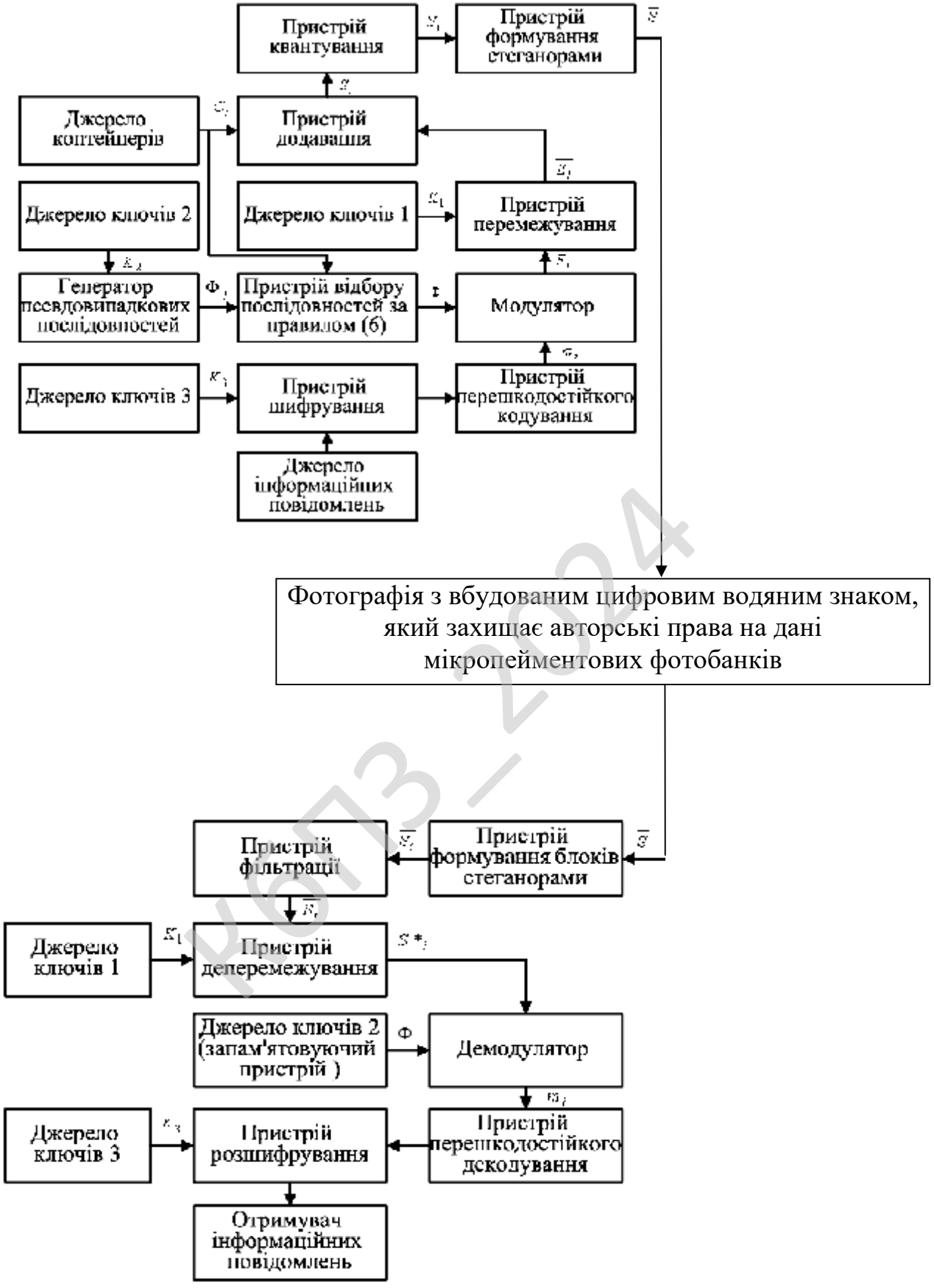

Рисунок 3.1 – Структурна схема системи

Якщо для сформованого дискретного сигналу Φ_j та для всіх $i = 0,..,N-1$ розраховані значення коефіцієнту кореляції $\rho(C_i, \Phi_j)$ менші або дорівнюють встановленому порогу $\rho_{\max}$, тобто, якщо виконується умова (3.14) для всіх блоків даних контейнеру, відповідне значення Φ_j приймається до подальшого стеганографічного приховування інформаційних даних. Сформовані таким чином псевдовипадкові послідовності складають ансамбль дискретних сигналів $\Phi = \{\Phi_0, \Phi_1, ..., \Phi_{M-1}\}$, вони враховують статистичні властивості контейнера та подаються до модулятора. На модулятор подається також блок інформаційних даних $m_i = (m_{i_0}, m_{i_1}, ..., m_{i_{k-1}})$, $k \le M$, який модулюється псевдовипадковими послідовностями за правилом (3.9). Сформоване таким чином модульоване повідомлення E_i подається на пристрій перемежування, ініційованого таємним ключем K_1, який сформовано джерелом ключів 1. Пристрій перемежування обробляє модульоване повідомлення E_i, тобто за правилом f, яке задає таємний ключ K_1, псевдовипадковим чином переставляє місцями елементи E_i. Отримані дані $\overline{E_i} = f(E_i, K_1)$ подаються на пристрій додавання, у якому виконується поелементне додавання з даними контейнеру C_i (з даними цифрового зображення в просторовій області): $S_i = C_i + \overline{E_i} \cdot G$, де $G > 0$ – коефіцієнт підсилення розширювального сигналу, який задає «енергію» вбудованих блоків інформаційного повідомлення. Контейнер формується джерелом контейнерів. Отримані дані S_i подаються на пристрій квантування, який виконує певне перетворення для зберігання початкового динамічного діапазону зображення-контейнеру, в результаті чого формуються окремі блоки стеганограми $\overline{S_i}$ та заповнений контейнер $\overline{S} = \overline{S_0} \cup \overline{S_1} \cup ... \cup \overline{S_{N-1}}$, який передається приймальній стороні.

Пристрій стеганографічного вилучення даних (див. рис. 3.1) працює аналогічно відповідному пристрою як і у відомому способі [1-3], за винятком генератора псевдовипадкових послідовностей, що формує відповідні дискретні сигнали. Пристрій функціонує наступним чином. Отримана стеганограмма $\overline{S}$ на

приймальній стороні подається на пристрій формування блоків стеганограми, в якому формуються блоки $\overline{S_i}$ та подаються на пристрій фільтрації. Після фільтрації отримані дані $\overline{\overline{S_i}}$ подаються на пристрій зворотного перемежування, на якому виконується дія, інверсна перемежуванню на передавальній стороні. Пристрій деперемежування ініційовано секретним ключем K_1, який сформовано джерелом ключів 1. Отримані після деперемежування дані S^*_i подаються на демодулятор, який виконує функцію кореляційного приймача дискретних сигналів за розглянутим вище правилом. Тобто в демодуляторі обчислюється значення коефіцієнту кореляції даних S^*_i та псевдовипадкових послідовностей Φ_i (дискретних сигналів). Ансамбль дискретних сигналів не формується (як у відомому способі) відповідним генератором псевдовипадкових чисел, що ініційований секретним ключем K_2, а зберігається у запам'ятовуючому пристрою, тобто весь ансамбль псевдовипадкових послідовностей $\Phi = \{\Phi_0, \Phi_1, ..., \Phi_{M-1}\}$ виступає у ролі секретного ключа K_2 і зберігається цілком в такому вигляді. Таким чином, запам'ятовуючий пристрій можна розглядати як аналог джерела ключів 2 у відомому способі [3], а послідовності, які надходять з нього є тотожними тим, які застосовуються на передавальній стороні при вбудовуванні інформаційних повідомлень. Таким чином, в демодуляторі обчислюється значення коефіцієнту кореляції між отриманими даними S^*_i та послідовностями, які застосовувалися при вбудовуванні інформації. Рішення, стосовно значення вбудованих даних, приймається відповідно до значення обрахованого коефіцієнту кореляції за правилом (5). Вилучені дані m_i подаються на пристрій перешкодостійкого декодування, в якому за визначеним правилом із використанням внесеної надмірності виправляються деякі помилки, відповідно до корегуючої здатності коду. Це призводить до деякого підвищення достовірності переданих даних, які після декодування подаються на пристрій розшифрування, ініційованого таємним ключем K_3, що формується джерелом ключів 3. Розшифровані повідомлення подаються отримувачу інформаційних повідомлень.

3.3 Розробка функціональної схеми

Технологія застосування системи припускає наявність мережі абонентів, що посилають один одному підписані фотографії з мікропейментового фотобанку. Для кожного абонента генерується пара ключів: закритий і відкритий.

Закритий ключ зберігається абонентом в таємниці і використовується їм для формування ЕЦП.

Відкритий ключ відомий всім іншим користувачам мікропейментового фотобанку і призначений для перевірки ЕЦП одержувачем підписаної фотографії з мікропейментового фотобанку. Відкритий ключ не дозволяє обчислити секретний ключ. Кожен легальний користувач може за допомогою відкритого ключа перевірити достовірність і незмінність електронного документу.

Цифровий водяний знак, прихований у файлі, служить гарантом того, що навіть якщо зловмисник підбере закритий ключ і підпише файл, результати перевірки підпису і ЦВЗ не співпадуть і можна буде встановити порушення. ЦВЗ виступає як додатковий рівень захисту, який іноді важко навіть виявити, а тим більше обійти.

Розроблена система складається з наступних модулів:

– генерація ключів (датчик псевдо-випадкових чисел);

– хеш-функція SHA-1;

– криптографічний модуль (алгоритми RSA та DES);

– стеганографічний модуль (для вбудовування ЦВЗ).

Модуль генерації ключів створює для користувача закритий та відкритий RSA ключі, та надсилає запит на сертифікацію відкритого ключа. У разі успішного пройдешня сертифікації, відкритий ключ, термін його придатності та інформація про власника зберігаються в базі даних (репозиторії), для того, щоб одержувачі підписаних документів могли перевірити його автентичність.

Хеш-функція та криптографічний модуль використовуються при створенні електронного цифрового підпису та при його перевірці. Також криптографічний

модуль задіяно при шифруванні та дешифруванні документу. Асиметричний алгоритм RSA використовується для створення підпису, а блочний алгоритм DES для шифрування.

Стеганографічний модуль необхідний для вбудовування цифрового водяного знаку в графічне зображення відсканованого документу та для його вилучення з документу і перевірки.

Центр сертифікації ключів (англ. Certification authority, CA) – це організація, яка надає послуги електронного цифрового підпису, зокрема:

– Надання у користування засобів електронного цифрового підпису.

– Допомога при генерації відкритих та особистих ключів.

– Обслуговування сертифікатів ключів:

– формування;

– розповсюдження ;

– скасування;

– зберігання;

– блокування;

– Надання інформації щодо чинних, скасованих і блокованих сертифікатів відкритих ключів.

– Послуги фіксування часу.

– Консультації та інші послуги.

Головним призначенням центру сертифікації є засвідчення автентичності відкритих ключів користувачів.

Центр сертифікації ключів, акредитований в установленому порядку, є акредитованим центром сертифікації ключів. Відмінністю акредитованого центру є те, що він має право обслуговувати виключно посилені сертифікати ключів.

Акредитований центр сертифікації ключів має виконувати усі зобов'язання та вимоги, встановлені законодавством для центру сертифікації ключів, та

додатково зобов'язаний використовувати для надання послуг електронного цифрового підпису надійні засоби електронного цифрового підпису.

Порядок акредитації та вимоги, яким повинен відповідати акредитований центр сертифікації ключів, встановлюються Кабінетом Міністрів України.

Відкриті ключі й інша інформація про користувачів зберігається центрами сертифікації, у вигляді цифрових сертифікатів, що мають наступну структуру:

– серійний номер сертифіката;

– об'єктний ідентифікатор алгоритму електронного підпису;

– ім'я центру, що засвідчує;

– строк придатності;

– ім'я власника сертифіката (ім'я користувача, якому належить сертифікат);

– відкритий ключ власника сертифіката;

– об'єктні ідентифікатори алгоритмів, асоційованих з відкритими ключами власника сертифіката.

Рисунок 3.2 – Функціональна схема роботи системи

Система включає дві процедури:

– процедуру захисту документу цифровими водяними знаками, електронним підписом та шифруванням;

– процедуру розшифрування та перевірки легітимності документу, шляхом перевірки ЕЦП та ЦВЗ.

У процедурі створення підпису використовується закритий ключ RSA відправника, в процедурі перевірки підпису – відкритий ключ RSA відправника. При вбудовуванні ЦВЗ в документ застосовується стегоключ, а при шифруванні секретний DES ключ, щоб забезпечити секретність цих ключів, вони шифруються відкритим ключем RSA одержувача, після чого лише він може розшифрувати їх своїм закритим RSA ключем.

Принциповим моментом в системі ЕЦП є неможливість підробки електронного підпису користувача без знання його секретного ключа.

Кожен підпис містить наступну інформацію:

– дату підпису;

– термін закінчення дії ключа даного підпису;

– інформацію про особу, що підписала файл (П.І.Б., посада, коротке найменування фірми);

– ідентифікатор того, хто підписав (ім'я відкритого ключа);

– власне цифровий підпис.

Перед відправкою файлу по мережі, в нього вбудовується ЦВЗ. При формуванні ЕЦП відправник перш за все обчислює хеш-функцію h(M) документу M, що слід підписати.

Обчислене значення хеш-функції h(M) являє собою один короткий блок інформації m, що характеризує весь документ M в цілому. Потім число m

шифрується секретним ключем відправника. Отримувана при цьому пара чисел є ЕЦП для даного документу М.

Далі документ шифрується і формується електронний конверт, що містить: зашифрований файл з попередньо вбудованими водяними знаками, підпис та зашифровані стего– та DES ключі. Електронний конверт передається по мережі, де може відбутися атака зловмисника на нього з метою підробки чи модифікації його змісту.

Прийнятий по каналу зв'язку документ М розшифровується, одержувач повідомлення, знову обчислює його хеш-функцію m=h(M), після чого за допомогою відкритого ключа відправника перевіряє, чи відповідає отриманий підпис обчисленому значенню m хеш-функції. Потім дешифрує своїм закритим ключем стегоключ, виймає з файлу стеговставку та порівнює її з шаблоном. Якщо перевірка підпису та цифрових водяних знаків пройшла успішно, то документ легітимний.

Хеш-функція призначена для стиснення підписуваного документа М до декількох десятків або сотень біт. Хеш-функція h() приймає як аргумент документ М довільного розміру і повертає хеш-значення h(M)=H фіксованої довжини. Зазвичай хеш-значення є стислим двійковим представленням основного повідомлення довільної довжини. Слід зазначити, що значення хеш-функції h(M) складним чином залежить від документа М і не дозволяє відновити сам документ М.

Хеш-функція повинна задовольняти цілому ряду умов:

– хеш-функція повинна бути чутлива до всіляких змін в тексті М, таких як вставки, викиди, перестановки і т.п.;

– хеш-функція повинна мати властивість безповоротності, тобто підбір документу М', який би мав необхідне значення хеш-функції, повинний бути неможливим;

– вірогідність того, що значення хеш-функцій двох різних документів (незалежно від їх довжин) співпадуть, повинна бути нікчемно малою.

Тобто хеш-функцією є таке математичне або алгоритмічне перетворення заданого блоку даних, яке володіє наступними властивостями:

– нескінченна область визначення;

– кінцева область значень;

– необоротність;

– зміна вхідного потоку інформації на 1 біт змінює близько половини всіх біт вихідного потоку.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. Після початку роботи розробленого ПЗ ми потрапляємо до головного вікна ПЗ далі проводяться дві основні дії.

Перша дія це наступна послідовність: формування контейнеру, обрання зображення; введення та формування повідомлення; генерація ключа контейнера; створення контейнеру з вбудованим повідомленням.

Друга дія: моніторинг наявності повідомлення; виведення результатів аналізу; відкриття зображення; дешифрування ключа; відкриття файлу з ключем; декодування повідомлення та виправлення помилок; виведення повідомлення на екран.

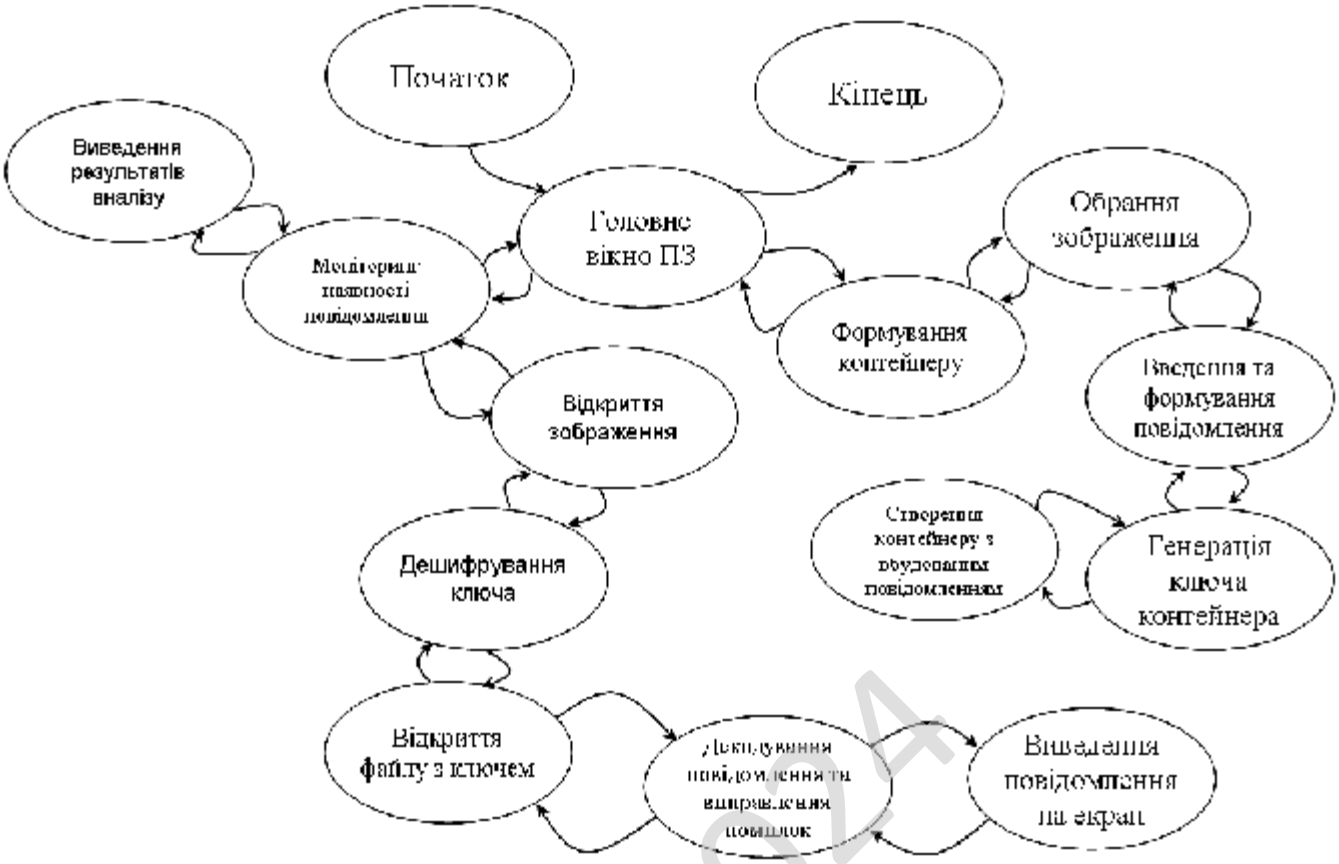

Рисунок 3.3 – Діаграма взаємодії процесів

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Розглянемо алгоритм роботи основної програми. Його блок-схема зображена на рисунку 4.1. З рисунку видно, що після запуску програми спочатку відбувається виведення вікна моніторингу програми. Потім здійснюється:

– Запит створення прихованого повідомлення (запит).

– Введення користувачем тексту повідомлення.

– Обрання зображення носія (контейнер).

– Генерація ключа контейнера.

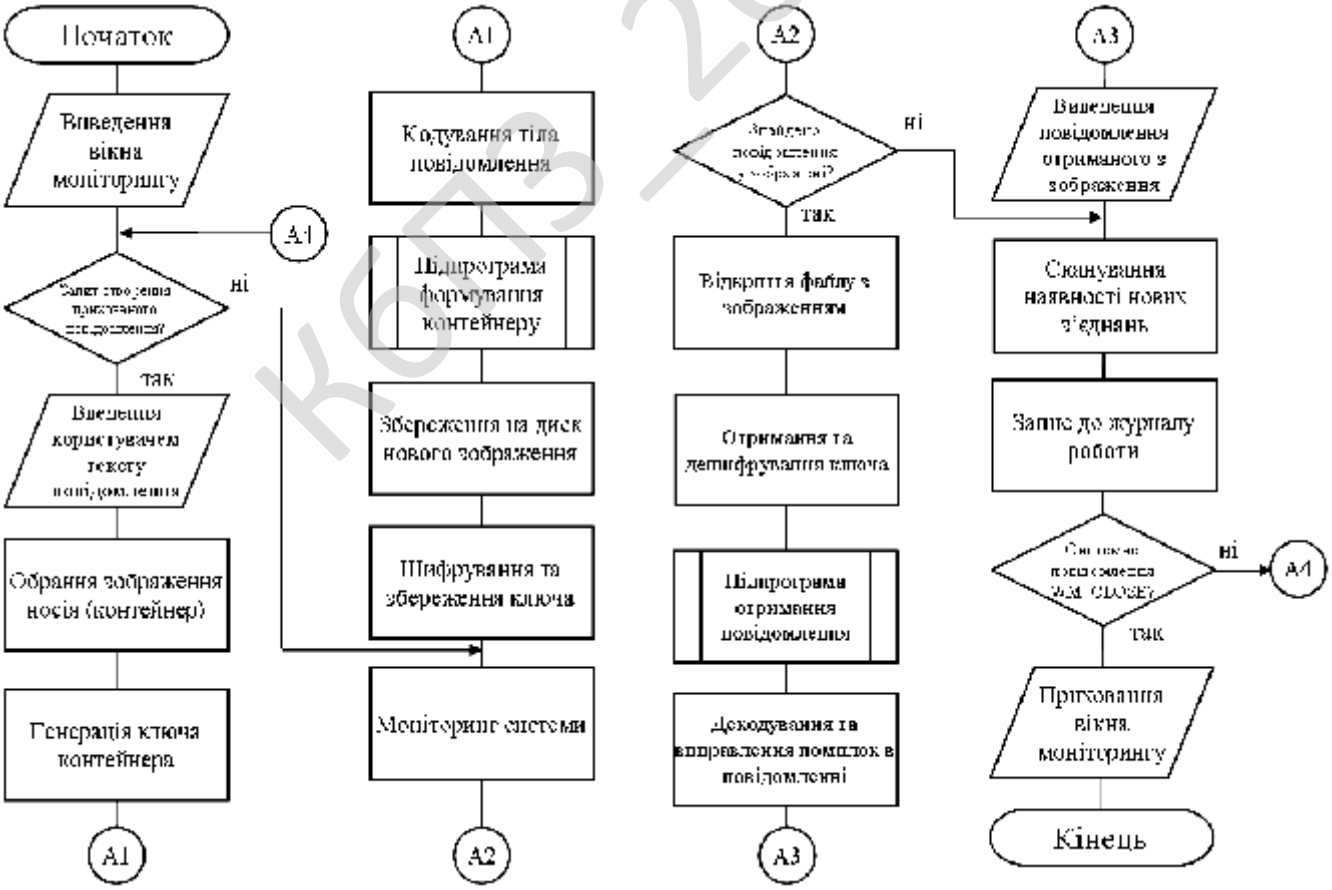

Рисунок 4.1 – Блок-схема основної програми

– Кодування тіла повідомлення.

– Підпрограма формування контейнеру.

– Збереження на диск нового зображення.

– Шифрування та збереження ключа.

– Моніторинг системи.

– Знайдено повідомлення у зображені (запит).

– Відкриття файлу з зображенням.

– Отримання та дешифрування ключа.

– Підпрограма отримання повідомлення.

– Декодування та виправлення помилок в повідомленні.

– Виведення повідомлення отриманого з зображення.

– Сканування наявності нових з’єднань.

– Запис до журналу роботи.

– Системне повідомлення WM_CLOSE (запит).

– Приховання вікна моніторингу.

На рисунку 4.2 зображена блок-схема роботи підпрограми формування контейнеру. Де відбуваються наступні дії:

– Читання тіла повідомлення.

– Формування сітки зображення.

– Створення масиву фрагментів зображення, скид лічильників у початковий стан.

– Читання значень R.G.B. поточного пікселя.

– Розрахунок скорегованих значень R.G.B. .

– Запит зменшити яскравість?.

– Збільшуємо яскравість пікселя.

– Запис змін до пікселя.

– Інкрементація лічильника фрагментів.

– Всі фрагменти оброблено (запит).

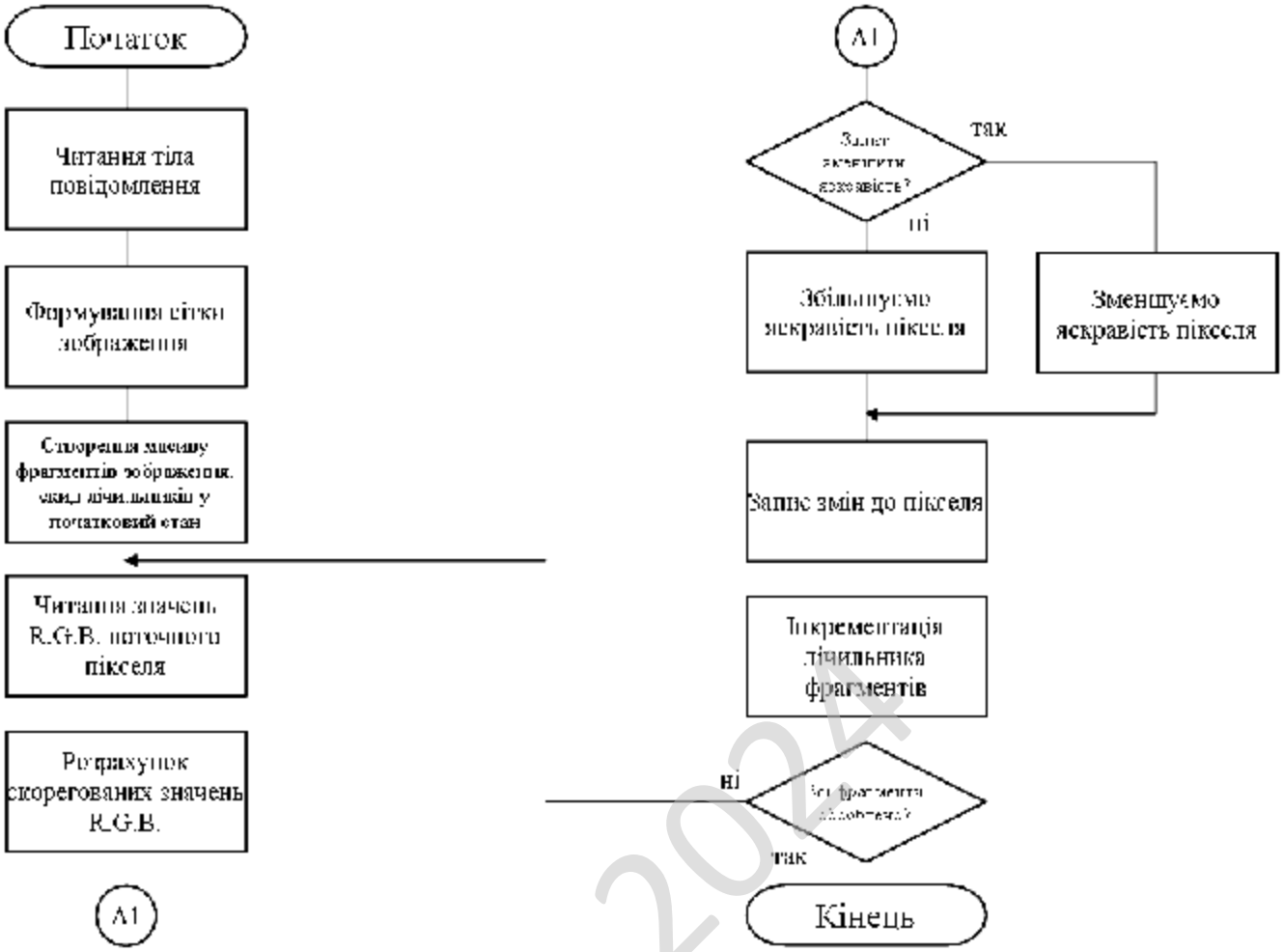

Рисунок 4.2 – Блок-схема підпрограми формування контейнеру

На рисунку 4.3 зображена блок-схема роботи підпрограми отримання повідомлення. Де відбуваються наступні дії:

– Відкриття знайденого зображення.

– Формування сітки зображення.

– Створення масиву фрагментів зображення, скид лічильників у початковий стан.

– Виділення пікселя з поточного фрагменту.

– Аналіз середнього значення яскравості інших пікселів у фрагменті.

– Вибраний піксель виділяється (запит).

– Додавання логічного нуля.

– Формування частини отриманого повідомлення.

– Інкрементація лічильника фрагментів.

– Всі фрагменти оброблено (запит).

– Запис журнальних даних до файлу.

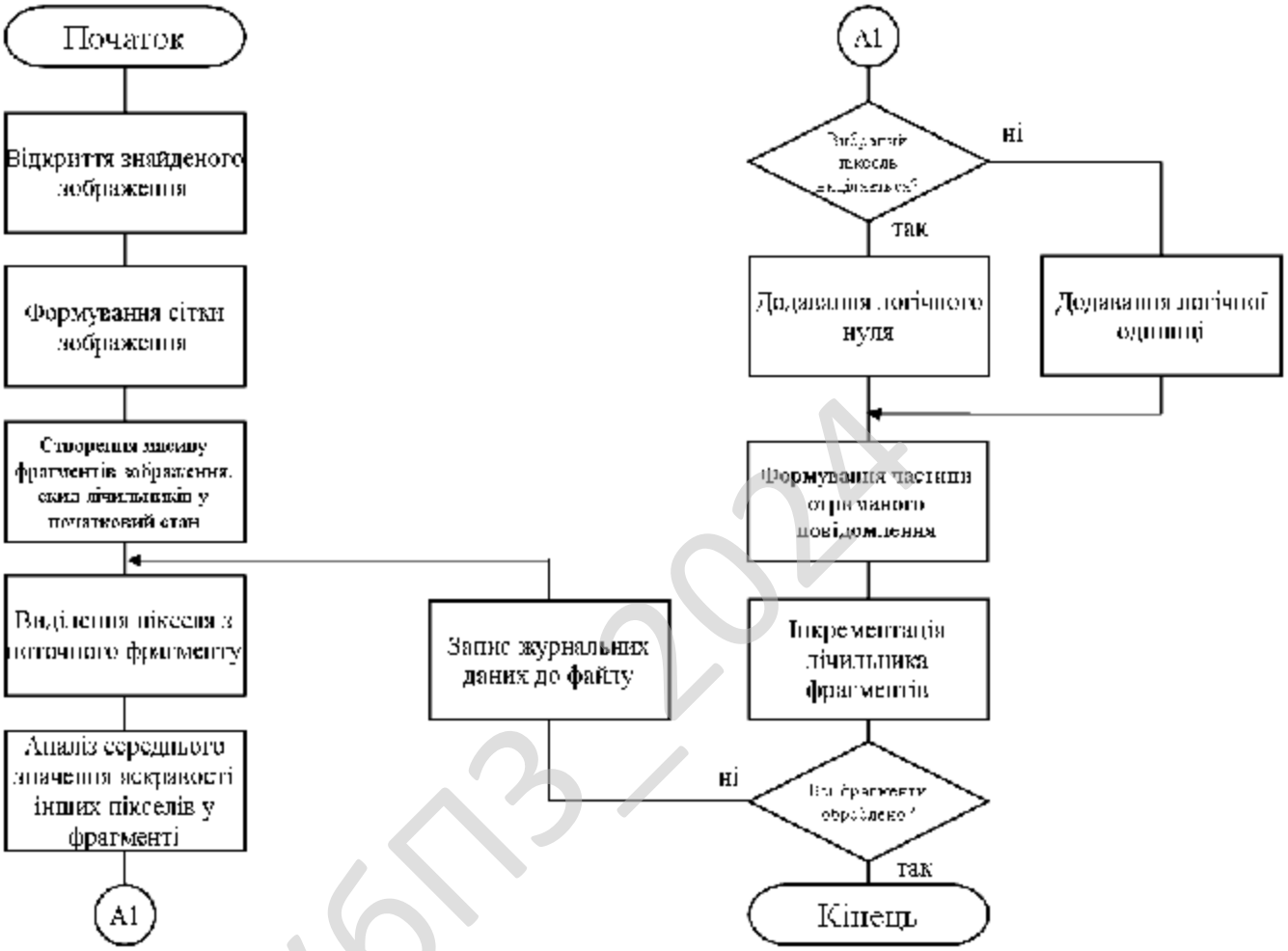

Рисунок 4.3 – Блок-схема підпрограми отримання повідомлення

Опис алгоритмів функціонування системи

Приведемо опис частини коду розробленого програмного забезпечення. Кидаємо на форму чотири кнопки, один мемо, один OpenPictureDialog, один OpenDialog і один SaveDialog.

```
var
IInfo:TImageFileInfo;
DFile:string;
```

Будемо там зберігати інформацію про файл, у якому будемо писати й шлях до поточного файлу даних. Одну кнопку назвемо так: “Відкрити файл-зображення”. В оброблювачі onClick пишемо код:

```
procedure TForm1.Button1Click(Sender: TObject);
var i:byte;
Strgs:TStringList;
begin
if OpenPictureDialog1.Execute then
   GetFileFormat(OpenPictureDialog1.FileName,Strgs,IInfo);
if OpenPictureDialog1.FileName<>'' then
begin
Memo1.Clear;
for i:=0 to Strgs.Count-1 do Memo1.Lines.Add(Strgs[i]);
end;
end;
```

Другу кнопку назвемо “Відкрити файл даних”. В оброблювачі пишемо наступне:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
if OpenDialog1.Execute then
DFile:=OpenDialog1.FileName;
end;
```

Третю кнопку назвемо “Сховати файл.” В оброблювачі:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
if DFile<>'' then IHide(IInfo,Dfile,SaveDialog1)
else MessageBox(handle,'Не обраний файл даних!!!','XData
      Error',MB_ICONERROR);
end;
```

А в оброблювачі четвертої, названої “Витягти з файлу”, напишемо тільки один рядок:

```
if OpenPictureDialog1.Execute then
    IExtract(OpenPictureDialog1.FileName,SaveDialog1);
```

Тепер створюємо скрин з екрана, відкриваємо його, обираємо файл, що хочемо сховати й ховаємо.

У файлі-контейнері ми будемо міняти останній (або декілька останніх) біт, у які й будемо записувати інформацію, яку треба сховати.

Спочатку прочитаємо заголовок bmp, визначимо – чи підходить він нам, потім перейдемо на тіло bmp, тобто на те, що містить сам малюнок.

Записувати будемо так: візьмемо розмір файлу, розіб'ємо на біти – запишемо, візьмемо довжину імені файлу, розіб'ємо на біти – запишемо, візьмемо саме ім'я файлу – розіб'ємо на біти, запишемо, потім уже записуємо саме тіло файлу.

Для початку напишемо процедуру, що буде читати заголовок bmp, а потім видавати нам всю потрібну інформацію про файл у вигляді запису.

Для початку визначимо тип запису:

```
type
TImageFileInfo=record
File:String;
// Ім'я bmp
width:dword;
// ширина рисунка
height:dword;
// висота малюнка
smeshenie:dword;
// Зсув малюнка від початку файлу
FileSize:dword;
// Розмір bmp
ImageSize:dword;
// Розмір малюнка
InfoHide:dword;
// Скільки інформації можна сховати
RLE:byte;
// Рівень стиску - повинен бути дорівнює 0
BitsPerPixel:byte;
// Розрядність малюнка (біт на піксель) - нам потрібні 24-х розрядні
Signature:string[2];
// Перші два байти файлу - bmp-файлів повинні бути 'BM'
end;
```

Тепер запишемо саму процедуру, що буде цей запис заповнювати. Крім того, вона буде відразу створювати StringList, у який буде поміщати опис файлу.

```
procedure GetFileFormat(File:string;var StrLst:TStringList;var
                        ImageInfo:TImageFileInfo);
var
```

```
Stream:TMemoryStream;
ch:char;
size,dwrd:dword;
wrd:word;
bt:byte;
str:string;
begin
StrLst:=TStringList.Create;
stream:=TMemoryStream.Create;
stream.LoadFromFile(File);
// Завантажуємо bmp у тільки що створений MemoryStream
stream.Read(ch,sizeof(ch));
// Читаємо сигнатуру
str:=ch;
stream.Read(ch,sizeof(ch));
str:=str+ch;
StrLst.Add(' ***********');
ImageInfo.Signature:=str;
// Занесемо в запис сигнатуру
if ImageInfo.Signature ='BM' then
// Якщо сигнатура = ‘BM’, те значить все нормально – продовжуємо
begin
ImageInfo.File:=File;
// Занесемо в запис ім'я bmp
stream.Read(dwrd,sizeof(dwrd));
// Уважаємо із заголовка розмір файлу
ImageInfo.Filesize:=dwrd;
size:=round(dwrd/1024);
// Переведемо в кілобайти для подальшого виводу в опис
StrLst.add('Розмір файлу із зображенням: '+IntToStr(dwrd)+'
          ( '+InTToStr(size)+' kb)');
stream.Read(dwrd,sizeof(dwrd));
// Зарезервовано – не використовується
stream.Read(dwrd,sizeof(dwrd));
// Зарезервовано – не використовується
ImageInfo.smeshenie:=dwrd;
// Якщо щось далі не ясно – дивися опис формату
StrLst.Add('Зсув даних бітового образа від заголовка:
           '+IntToStr(dwrd)+' байт');
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Розмір BITMAPINFOHEADER: '+IntToStr(dwrd)+' байт');
stream.Read(dwrd,sizeof(dwrd));
```

```
ImageInfo.width:=dwrd;
StrLst.Add('Ширина зображення: '+IntToStr(dwrd)+' пікселів');
stream.Read(dwrd,sizeof(dwrd));
ImageInfo.height:=dwrd;
StrLst.Add('Висота зображення: '+IntToStr(dwrd)+' пікселів');
stream.Read(wrd,sizeof(wrd));
StrLst.Add('Число бітових площин пристрою: '+IntToStr(wrd));
stream.Read(wrd,sizeof(wrd));
ImageInfo.BitsPerPixel:=wrd;
StrLst.Add('Глибина зображення (число бітів на піксель): '+IntToStr(wrd));
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Тип стиску ( 0-відсутній): '+IntToStr(dwrd));
stream.Read(dwrd,sizeof(dwrd));
size:=round(dwrd/(1024));
StrLst.Add('Розмір картинки в байтах: '+IntToStr(dwrd)+' (
          '+IntToStr(size)+' kb)');
ImageInfo.ImageSize:=dwrd;
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Горизонтальний дозвіл пристрою (піксель/м): '+IntToStr(dwrd));
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Вертикальний дозвіл пристрою (піксель/м): '+IntToStr(dwrd));
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Число використовуваних квітів: '+IntToStr(dwrd));
stream.Read(dwrd,sizeof(dwrd));
StrLst.Add('Число важливих квітів: '+IntToStr(dwrd));
ImageInfo.InfoHide:=round((ImageInfo.width*ImageInfo.height*3)/8);
StrLst.Add(' **********');
StrLst.Add('');
StrLst.Add('Можна сховати інформації '+IntToStr(ImageInfo.infohide)+'байт
          ('+IntToStr(round((ImageInfo.infohide)/1024))+'kb)');
StrLst.Add('');
end
else
StrLst.Add('Помилка: це не файл формату BMP.');
stream.Free;
end;
```

Тепер займемося написанням процедури, що буде розбивати байт на біти, і видавати результат у вигляді масиву 1 і 0.

Для цього скористаємося логічною командою AND.

Вираження істинно, коли щирі обидва біти. А так само будемо використовувати SHL – логічне зрушення вліво, операція застосовується тільки до групи бітів, одного із цілочислених типів Паскаля, наприклад до байта, слова й т.д.

Зрушення байта вліво на один розряд. Байт зміщається вліво на один або більше розрядів, позиції праворуч заміщаються нулями, позиції ліворуч губляться. Ми будемо це використовувати, щоб циклічно порівнювати кожний біт з одиницею.

Тобто ми будемо поступово зрушувати одиницю вліво, а всі інші позиції будуть рівні 0. Це буде виглядати так: якщо SByte and (1 shl num)<>0, то в байті SByte – біт з номером num = 1, інакше цей біт = 0. Поставивши цикл for від 0 до 7 – ми переберемо весь байт, і створимо масив з 1 і 0.

```
type
BitsArray=array[0..7] of byte;
procedure HexToBin(x:byte;var a:BitsArray);
var
bit,i:byte;
begin
for i:=7 downto 0 do
begin
if (X and (1 shl i))<>0 then bit:=1 else bit:=0;
a[i]:=bit;
end;
end;
```

Тепер напишемо процедуру для зворотної дії, тобто збору байта з масиву 1 і 0.

```
procedure BinToHex(var x:byte;a:BitsArray);
var i:dword;
bit,t:byte;
begin
x:=0;
for i:=7 downto 0 do
begin
bit:=a[i];
if (bit and (1 shl 7)) <> 0 then bit:=1;
x:=x or bit;
```

```
if i<>0 then x:=x shl 1;
end;
end;
```

Тепер напишемо дві процедури для встановлення/скидання біта певного номера.

```
function SetBit(SByte: byte; num: byte): byte;
begin
Result := SByte or (1 shl num);
end;
function ResetBit(SByte: byte; num: byte): byte;
begin
Result := SByte and not (1 shl num);
end;
```

Тут SByte – вихідний байт, а num – номер біта, якому треба скинути. Прошу врахувати, що біти природно йдуть від нульового номера й до сьомого.

А тепер перейдемо до самого цікавого – до написання процедури, що буде ховати файл-даних у файл-контейнер (bmp). Відразу поясню загальну логіку.

Спочатку ми одержуємо інформацію про bmp (GetFileFormat), потім дивимося: якщо в дану bmp влізе файл такого обсягу, то продовжуємо. Вантажимо обидва файли на згадку за допомогою TMemoryStream.

Потім в bmp переходимо на тіло малюнка, тобто робимо в memorystream'е position:=ImageFile.Smeshenie. Далі беремо розмір файлу даних (dword, тобто 4 байти, тобто 32 біта) і записуємо в 32 байта bmp, скрізь міняючи біт самого дрібного розряду.

Тобто ми максимум змінимо значення байта кольору в малюнку на ±1, що візуально зовсім не помітно (дослідним шляхом перевірено що міняти можна й 2 біти, візуально не міняючи картинку). Далі записуємо у файл-контейнер довжину імені файлу-даних.

Виходячи з того що довжина буде <=255 символам ми відводимо під неї один байт, тобто міняємо останній біт в 8 байтах bmp. Далі записуємо ім'я файлу контейнера.

Для цього нам знадобляться два цикли: один зовнішній, для того щоб пройтися по кожному символі ім'я, а інший внутрішній, котрий буде записувати

кожний символ у вигляді 8 битв bmp. Далі – записуємо саме тіло файлу контейнера.

Послідовно розбиваємо кожний байт на біти й пишемо.

```
procedure IHide(ImageInfo:TImageFileInfo; DFile:string;
          SaveDialog:TSaveDialog);
var
sz,n,z:dword;
len,v,i,xi,xd:byte;
FIStream,FDStream:TMemoryStream;
BAr,BArLog:BitsArray;
tmp,FName:string;
f:File;
begin
// В FDStream довантажується файл даних, а в FIStream файл-контейнер
FDStream:=TMemoryStream.Create;
FDStream.LoadFromFile(DFile);
FIStream:=TMemoryStream.Create;
// перевірки - підходить цей файл на роль контейнера чи ні
if ImageInfo.Signature<>'BM' then
begin
MessageBox(Application.handle,'Це не зображення формату Windows
           Bitmap.','Error',MB_ICONERROR);
exit;
end;
if ImageInfo.BitsPerPixel<>24 then
begin
MessageBox(Application.handle,'Зображення повинне бути 24-х
          бітним!','Error',MB_ICONERROR);
exit;
end;
if ImageInfo.RLE<>0 then
begin
MessageBox(Application.handle,'У даному зображенні використовується
          стиск.','Error',MB_ICONERROR);
exit;
end;
if (ImageInfo.InfoHide-300) begin
MessageBox(Application.handle,'Це зображення занадто мало, щоб помістити в
           себе стільки інформації.','Error',MB_ICONERROR);
exit;
end;
```

```
// Якщо файл пройшов всі перевірки - довантажуємо його на згадку
FIStream.LoadFromFile(ImageInfo.File);
FIStream.Position:=ImageInfo.smeshenie;
// Розмір файлу даних поміщаємо в sz
sz:=FDStream.size;
// Записуємо розмір файлу який ховаємо в bmp
for n:=31 downto 0 do
begin
// Читаємо байт із файлу-контейнера
FIStream.Read(xi,sizeof(xi));
// Якщо в sz'е біт з номером n дорівнює 1,
// то встановлюємо 0-ой біт у тільки що зчитаному
// байте, ну а якщо n'ный біт скинутий, те скидаємо 0-ой біт...
if (sz and (1 shl n))<>0 then xi:=SetBit(xi,0)
else xi:=ResetBit(xi,0);
// Відкочуємося на позицію назад, щоб
// перезаписати той біт який ми вважали, але
// тільки тепер зі зміненим нульовим бітом
FIStream.Position:=FIStream.Position-1;
FIStream.Write(xi,sizeof(xi));
end;
// Одержуємо ім'я файлу даних без шляху до файлу
tmp:=ExtractFileName(DFile);
// Одержуємо довжину імені
len:=Length(tmp);
// Розбиваємо байт у якому зберігається довга ім'я на масив 1 і 0
HexToBin(len,BAr);
// Записуємо довжину імені у файл-контейнер
for n:=7 downto 0 do
begin
FIStream.Read(xi,sizeof(xi));
if BAr[n]=1 then xi:=SetBit(xi,0);
if BAr[n]=0 then xi:=ResetBit(xi,0);
FIStream.Position:=FIStream.Position-1;
FIStream.Write(xi,sizeof(xi));
end;
// Записуємо саме ім'я файлу в bmp
for n:=1 to len do
begin
// Беремо символ з номером n, одержуємо його
// ascii-код, розбиваємо отримане на біти й // записуємо
v:=ord(tmp[n]);
```

```
HexToBin(v,BAr);
for i:=7 downto 0 do
begin
FIStream.Read(xi,sizeof(xi));
if BAr[i]=1 then xi:=SetBit(xi,0);
if BAr[i]=0 then xi:=ResetBit(xi,0);
FIStream.Position:=FIStream.Position-1;
FIStream.Write(xi,sizeof(xi));
end;
end;
// Пишемо в bmp саме тіло файлу
// кожний байт розбиваємо на біти й записуємо
for n:=0 to ( sz-1) do
begin
FDStream.Read(xd,sizeof(xd));
HexToBin(xd,BAr);
for i:=7 downto 0 do
begin
FIStream.Read(xi,sizeof(xi));
if BAr[i]=1 then xi:=SetBit(xi,0);
if BAr[i]=0 then xi:=ResetBit(xi,0);
FIStream.Position:=FIStream.Position-1;
FIStream.Write(xi,sizeof(xi));
end;
end;
SetLength(FName,0);
// Викликаємо... ні, не парфумів, а діалог збереження файлів
if SaveDialog.Execute then FName:=SaveDialog.FileName;
// Зберігаємо
FIStream.SaveToFile(FName);
// Звільняємо змінні
FDStream.Free;
FIStream.Free;
ShowMessage('Done!');
end;
```

Напишемо процедуру, що це все буде витягати з картинки.

```
procedure IExtract(IFile:string;SaveDialog:TSaveDialog);
var
IInfo:TImageFileInfo;
Lst:TStringList;
IFstream,DFStream:TMemoryStream;
```

```
n,i,sz:dword;
a:BitsArray;
b:array[0..31] of byte;
bit,len,x,z:byte;
DFile:string;
begin
Lst:=TStringList.Create;
// Одержуємо інформацію з bmp
GetFileFormat(IFile,Lst,IInfo);
// Вантажимо все на згадку
IFstream:=TMemoryStream.Create;
DFStream:=TMemoryStream.Create;
IFstream.LoadFromFile(IFile);
IFstream.Position:=IInfo.smeshenie;
// Збираємо масив з 1 і 0, що є розмір захованого файлу
for n:=31 downto 0 do
begin
IFstream.Read(z,sizeof(z));
if (z and 1)<>0 then z:=1 else z:=0;
b[n]:=z;
end;
sz:=0;
// Збираємо dword по бітах з масиву отриманого вище
for i:=31 downto 0 do
begin
bit:=b[i];
if (bit and (1 shl 7)) <> 0 then bit:=1;
sz:=sz or bit;
if i<>0 then sz:=sz shl 1;
end;
SetLength(DFile,0);
DFIle:='';
z:=0;
// Одержуємо з файлу-контейнера довжину імені
for n:=7 downto 0 do
begin
IFstream.Read(z,sizeof(z));
if (z and 1)<>0 then z:=1 else z:=0;
a[n]:=z;
end;
len:=0;
// Масив 1 і 0-0->байт
```

```
BinToHex(len,a);
// Одержуємо ім'я захованого файлу
for n:=1 to len do
begin
for i:=7 downto 0 do
begin
IFstream.Read(z,sizeof(z));
//  Error!
if (z and 1)<>0 then z:=1 else z:=0;
a[i]:=z;
end;
BinToHex(z,a);
DFile:=Dfile+chr(z);
end;
// Дістаємо саме тіло файлу
for i:=0 to sz-1 do
begin
// Зчитуємо 8 біт
for n:=7 downto 0 do
begin
IFstream.Read(z,sizeof(z));
if (z and 1)<>0 then z:=1 else z:=0;
a[n]:=z;
end;
// Збираємо із цих біт байт
BinToHex(x,a);
// Дописуємо байт до тому що є в пам'яті - збираємо файл
DFStream.Write(x,sizeof(x));
end;
try
// Запропонуємо для збереження файлу ім'я, під яким він був упакований.
SaveDialog.FileName:=DFile;
// Викликаємо діалог збереження
if SaveDialog.Execute then DFile:=SaveDialog.FileName;
// Зберігаємо витягнутий файл
DFStream.SaveToFile(DFile);
except
MessageBox(0,PAnsiChar('Помилка при спробі збереження.'+#13#10+DFile),' X-
          Date',MB_ICONERROR);
end;
IFstream.Free;
DFStream.Free;
```

```
Lst.Free;
MessageBox(0,'Done!',' X-Data',MB_ICONINFORMATION);
end;
```

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм SEED – у криптографії симетричний блоковий криптоалгоритм на основі Мережі Фейстеля, розроблений Корейським агентством інформаційної безпеки (Korean Information Security Agency, KISA) в 1998 році. В алгоритмі використовується 128-бітний блок і ключ довжиною 128 біт. Алгоритм одержав широке поширення й використовується фінансовими й банківськими структурами, виробничими підприємствами й бюджетними установами Південної Кореї, оскільки 40-бітний SSL не забезпечує на даний момент мінімально необхідного рівня безпеки. Агентством по захисту інформації специфіковане використання шифру SEED у протоколах TLS і S/MIME. У той же час, алгоритм SEED не реалізований у більшості сучасних браузерів і інтернет-додатків, що утрудняє його використання в даній сфері поза межами Південної Кореї.

SEED являє собою мережу Фейстеля з 16 раундами, 128-бітовими блоками й 128-бітовим ключем. Алгоритм використовує дві 8×8 таблиці підстановки, які, як такі з Safer, виведені з дискретного зведення в ступінь (у цьому випадку, x^{247} і x^{251} – плюс деякі «несумісні операції»). Це є деякою подібністю с MISTY1 у рекурсивності його структури: 128-бітовий повний шифр – мережа Фейстеля з F-функцією, що впливає на 64-бітові половини, у той час як сама F-функція – Мережа Фейстеля, складена з G-функції, що впливає на 32-розрядні половини. Однак рекурсія не простягнеться далі, тому що G-функція – не Мережа Фейстеля. В G-функції 32-розрядне слово розглядають як чотири 8-бітових байта, кожний з яких проходить через одну або іншу таблицю підстановки, потім поєднується в

помірковано комплексному наборі булевих функцій таким чином, що кожний біт виводу залежить від 3 з 4 вхідних байтів.

SEED має складний ключовий розклад, генеруючи тридцять два 32-розрядних додаткових символу, використовуючи G-функції на серіях обертань вихідного неопрацьованого ключа, комбінованого зі спеціальними раундовими константами (як в TEA) від «Золотого співвідношення» (англ. Golden ratio).

Згідно з дослідженнями KISA, алгоритм SEED «надійно протистоїть відомим атакам».

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ КІБЕРБЕЗПЕКИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

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

З нього видно, що інтерфейс користувача програми складається з таких логічних блоків:

– Меню: Дані; Шаблони; Налаштування; Довідка.

– Функцій програми: Обробка; Моніторинг; Завантажити зображення; Сформувати тіло повідомлення; Шаблони; Ключі; Аналіз зараження; Налаштування.

– Вікно відображення дій над зображенням: вилучення / додавання.

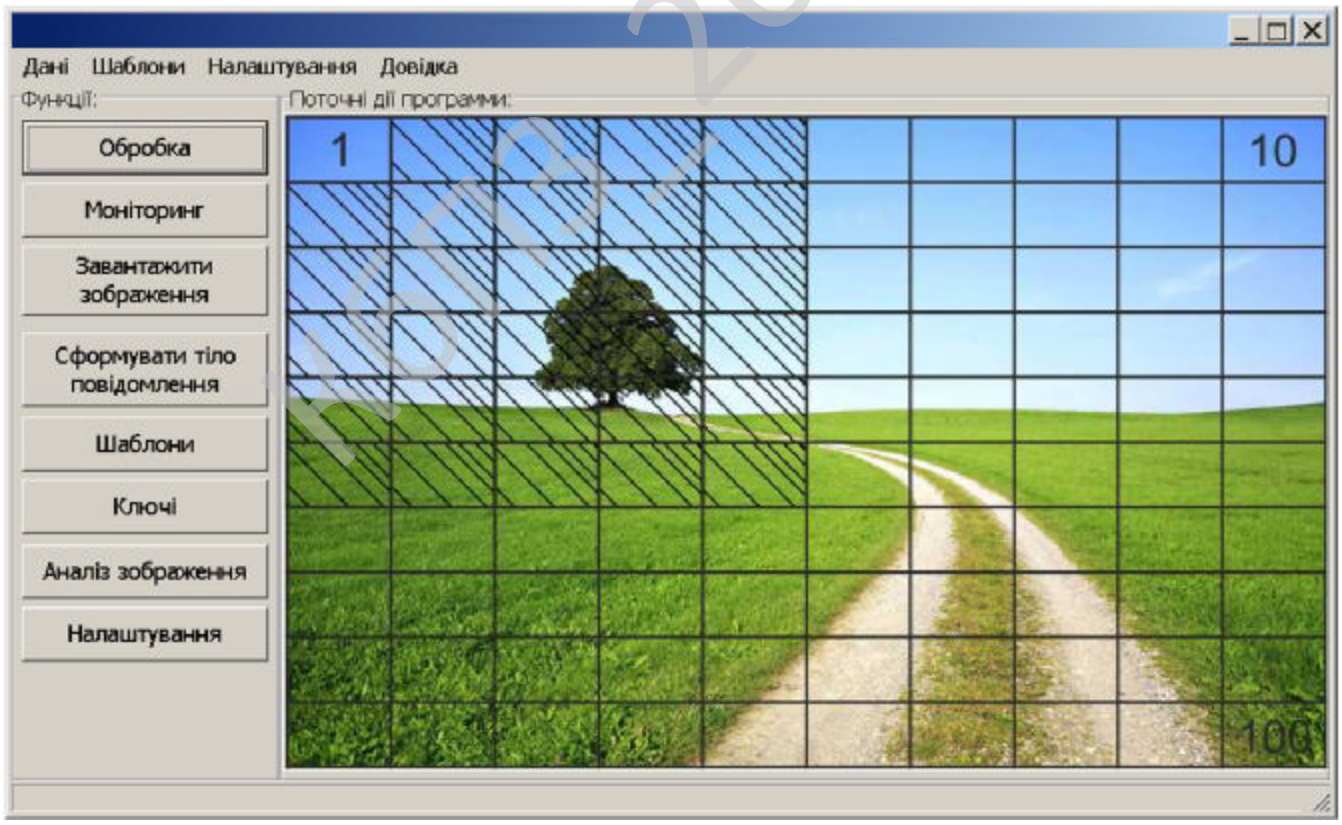

Рисунок 5.1 – Головне вікно програми

На рисунку 5.2 зображено форму авторського права, де відображені дані розробника.

Було обрано Shareware умову розповсюдження. Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (неповнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми.

В угоді про використання (ліцензії для кінцевого користувача, EULA) також може бути обумовлена заборона на комерційне або професійне (не тестове) її використання. Основний принцип умовно-безплатного ПЗ – «спробуй, перш ніж купити» (try before you buy). ПЗ що поширюється як умовно-безплатний, надається користувачам безоплатно.

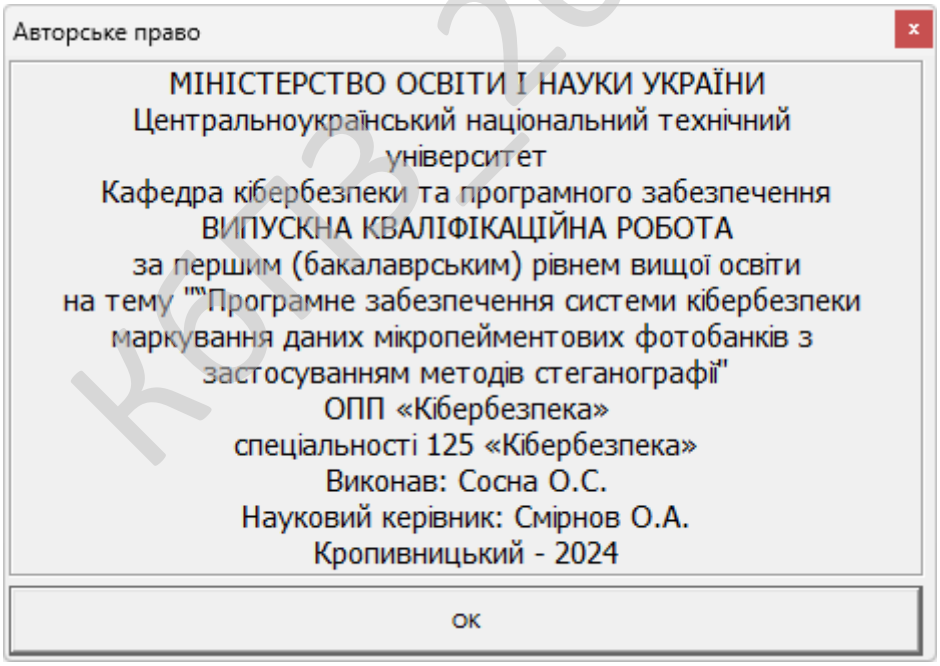

Рисунок 5.2 – Форма авторського права

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти, призначено для системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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 маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

– Досліджена система маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

– На основі отриманих результатів досліджень створена програмна реалізація системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії.

Розроблені під час виконання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання маркування даних мікропейментових фотобанків з застосуванням методів 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 Sydney. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для

системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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 SEED.

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. Derek Fisher. Application Security Program Handbook. Manning Publications. 2021. 155 p.

2. Cameron Wyatt PH.D. Kali Linux Tutorial. Independently published. 2021. 60 p.

3. Alex Matrosov, Eugene Rodionov, Sergey Bratus. Rootkits and Bootkits. No Starch Press. 2019. 450 p.

4. Kuznetsov, O., Frontoni, E., Kandiy, S., Smirnova, T., Prokopov, S., Bilanovych, A. «New Cost Function for S-boxes Generation by Simulated Annealing Algorithm». Lecture Notes on Data Engineering and Communications Technologies, 2023. vol 180. pp. 310-320. Springer, Cham.

5. Kuznetsov, O., Frontoni, E., Kandiy, S., Smirnov, O., Ulianovska, Y., Kobylianska, O. «Heuristic Search for Nonlinear Substitutions for Cryptographic Applications». *Lecture Notes on Data Engineering and Communications Technologies,* 2023. vol 180. Springer, Cham. pp. 288-298.

6. Kuznetsov, O., Kuznetsova, Y., Smirnov, O., Kostenko, O., Zvieriev, V. «Evaluating Hashing Algorithms in the Age of ASIC Resistance». *CEUR Workshop Proceedings*, 2023, 3628, pp. 93-105.

7. Kuznetsov O., Frontoni E., Kuznetsova Ye., Smirnov O., Chevardin V. «Achieving Enhanced Security in Biometric Authentication: A Rigorous Analysis of Code-Based Fuzzy Extractor». *CEUR Workshop Proceedings,* Volume 3624, 2023, pp. 330-339.

8. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

9. Kuznetsov, O., Kandiy, S., Frontoni, E., Smirnov, O. «Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece». *CEUR Workshop Proceedings,* Volume 3504, 2023, pp. 1-11.

10. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» *CEUR Workshop Proceedings,* Volume 3187, 2022,

11. Smirnov, O., Lakhno, V., Akhmetov, B., Chubaievskyi, V., Khorolska, K., Bebeshko, B. «Selection of a Rational Composition of İnformation Protection Means Using a Genetic Algorithm». *In: Rajakumar, G., Du, KL., Vuppalapati, C., Beligiannis, G.N. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. Lecture Notes on Data Engineering and Communications Technologies*, vol 131. 2023. Springer, Singapore. pp. 21-34.

12. Smirnov O.A., Al-Oraiqat A.M., Ulichev O.S., Meleshko Ye.V., Al-Rawashdeh H.S., Polishchuk L.I. «Modeling strategies for information influence dissemination in social networks». *Journal of Ambient Intelligence and Humanized Computing* Volume 13, Issue 5. Springer, Cham. 2022, pp. 2463-2477.

13. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». *11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021,* Cracow, Poland, 22-25 September 2021. P. 414-418

14. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». *4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) - 2021*, Lviv, Ukraine, September 21-25, 2021. P. 255-260.

15. Smirnov O., Kuznetsov A., Girzheva O., Kiian A., Nakisko O., Kuznetsova T. «Advanced Code-Based Electronic Digital Signature Scheme». *2020 IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020,* Kharkiv, 6 October 2020-9 October 2020, P. 358-362.

16. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova K. «Data hiding scheme based on spread sequence addressing». *CEUR Workshop Proceedings* Volume 2805, 2020, Pages *44-58.*

17. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». *International Journal of Computing*; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.

18. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

19. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

20. Smirnov O., Kuznetsov A., Arischenko A., Chepurko I., Onikiychuk A., Kuznetsova T. «Pseudorandom sequences for spread spectrum image steganography». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 122-131.

21. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». *CEUR Workshop Proceedings* Volume 2654, 2020, Pages 1-14.

22. Smirnov O., Lutsenko M., Kuznetsov A., Kiian A., Kuznetsova T., «Biometric cryptosystems: overview, state-of-the-art and perspective directions». *Lecture Notes in Networks and Systems*, vol 152. Springer, Cham. 2021, pp 66-84.

23. Smirnov O., Kuznetsov A., Onikiychuk A., Makushenko T., Anisimova O., Arischenko A. «Adaptive pseudo-random sequence generation for spread spectrum image steganography». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Ukraine, Kyiv, May 14-18. 2020. P. 161-165.

24. Smirnov O., Kuznetsov A., Kiian A., Babenko V., Perevozova I.,Chepurko I. «New Approach to the Implementation of Post-Quantum Digital Signature Scheme». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 166-171.

25. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

26. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

27. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

28. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». *International Journal of Computer Network and Information Security (IJCNIS)*. Vol. 12, No. 3, 2020. PP.33-43.

29. Smirnov, O., Kuznetsov, A., Gorbacheva, L., Babenko, V., «Hiding data in images using a pseudo-random sequence», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 646-660.

30. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

31. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of

Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

32. Smirnov, O., Ulichev, O., Meleshko, Y., Khokh, V., Goncharenko, I. «Method of Choosing Objects for Informational Influence in Social Networks during Information Campaign Based on the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, Vol 2588, P. 215-227, 2019.

33. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

34. Smirnov, O., Kuznetsov, A., Kiian, A., Gorbenko, Y., Cherep, O., Bexhter L. «Code-based Pseudorandom Generator for the Post-Quantum Period», *2019 IEEE International Conference on Advanced Trends in Information Theory (IEEE ATIT 2019).* 18.12.19-20.12.19 Kyiv Ukraine. P. 204 – 209.

35. Smirnov, O., Kuznetsov, A., Nariezhnii, O., Stelnyk, S., Kokhanovska, T., Kuznetsova T., «Side Channel Attack on a Quantum Random Number Generator», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18 - 21 September 2019. P.713-718.

36. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019;* Metz; France; 18-21 September 2019. P. 707-712.

37. Smirnov, O., Kuznetsov, A., Stefanovych, O., Gorbenko, Y., Krasnobaev, V., Kuznetsova K. «Information Hiding Using 3D-Printing Technology», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.701-706.

38. Smirnov, O., Hu, Z., Vasiliu, Y., Sydorenko, V., Polishchuk, Y., «Abstract Model of Eavesdropper and Overview on Attacks in Quantum Cryptography Systems», *10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019*; Metz; France; 18-21 September 2019. P.399-405.

39. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

40. Smirnov, O., Kuznetsov, A., Kiian, A., Babenko, B., Zhosan, H., Prokopovych-Tkachenko, D., «Soft Decoding Method for Turbo-Productive Codes», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT 2019*, Lviv, Ukraine, 2-6 July, 2019, P. 129-134.

41. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

42. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

43. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

44. Smirnov, O., Kuznetsov, A., Kiian, A., Kuznetsova, K., Ivko, T., Prokopovych-Tkachenko, D., «Soft Decoding Based on Ordered Subsets of Verification Equations of Turbo-Productive Codes», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 873-884.

45. Smirnov A.A., Kuznetsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

46. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». *Кібербезпека: освіта, наука, техніка.* 2024. №3(23), С. 111-131.

47. Смірнов О.А. Козлов Я.О., Смірнова Т.В. «Дослідження застосування SIEM-систем для забезпечення кібербезпеки та захисту інформації». *II Міжнародна науково-практична Інтернет-конференція «Інновації та перспективні шляхи hозвитку інформаційних технологій (ІПШРІТ-2023)»* м.Черкаси 6 грудня 2023 року – Черкаси: ЧДТУ.– 2023. – С.251-252.

48. Козлов Я.О., Смірнова Т.В., Смірнов О.А. «Дослідження SIEM-систем для забезпечення кібербезпеки». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя кафедри кібербезпеки та програмного забезпечення,* м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 26.

49. Козлов Я.О., Козірова Н.Л., Смірнов О.А. «Дослідження структури та принципу роботи SIEM-системи». *VII міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології" до 30-ти річчя кафедри кібербезпеки та програмного забезпечення,* м. Кропивницький. 1 листопада 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 59.

50. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». *Системи управління, навігації та зв'язку*, 2023, вип. 2(72), С. 170-178.

Додаток А
(обов’язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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. № 135-02 від 01.04.2024 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти є розробка програмного забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів 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 маркування даних мікропейментових фотобанків з застосуванням методів 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, 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 Sydney.

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 – 78 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 23.05.2024 р.

9.2 Подання випускної кваліфікаційної роботи за першим (бакалаврським) рівнем вищої освіти на захист 7.06.2024 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

_____ Смірнов О.А.

Програмне забезпечення системи кібербезпеки маркування даних мікропейментових фотобанків з застосуванням методів стеганографії

Лістинг програми

Код документу 12

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

Загальна кількість аркушів: 28

Літера: РП

Кропивницький – 2024 року

PASSWORD__STEGOGRAPHY.PAS - модуль створення стегоключа

```
unit password__Stegography;

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.
```

HAMMING.PAS – перешкодостійке кодування методом Хеммінга

```
unit Hamming;

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 Хеммінга
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.
```

DES.PAS – шифрування стегоключа алгоритмом DES

```
unit DES;

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
DES_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);

DES_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);

DES_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);

DES_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))
);

DES_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);

DES_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);

DES_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 Dest:TBitString; Source:TBitString; NBits:Integer);
```

```
Function ConcatBits(Bits:Array Of TBitString):TBitString;

Function DESEncode(S,Key:String):TBitString;
Function DESDecode(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 Dest:TBitString; Source:TBitString; NBits:Integer);
Var
I:Integer;
Begin
SetLength(Dest,NBits);
For I:=0 To NBits-1 Do
  Dest[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(DES_PC1));
For I:=0 To Length(DES_PC1)-1 Do
  Result[I]:=Key[DES_PC1[I]-1];
End;

Function GetPermutedKey2(Key:TBitString):TBitString;
Var
I:Integer;
Begin
SetLength(Result,Length(DES_PC2));
For I:=0 To Length(DES_PC2)-1 Do
  Result[I]:=Key[DES_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,DES_LSH[ I-1]);
  Result[I].D:=LeftShift(Result[ I-1].D,DES_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(DES_IP)-1 Do
  IP[I]:=M[DES_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[DES_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[DES_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(DES_REVERSE_IP)-1 Do
  Result[I]:=RL[DES_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 DESEncode(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 DESDecode(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.PAS – основна программа

```
unit main_Stegography;

interface

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

type

  T_Stegography = 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_StegographyMenu1: TMain_StegographyMenu;
    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: T_Stegography;

implementation

uses password__Stegography, Hamming;

{$R *.DFM}

// завантаження зображення
procedure T_Stegography.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.FormCreate(Sender: TObject);
begin
   loaded_:=false;
end;

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

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

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

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

end.
```

STEGOGRAPHY.PAS – реалізація алгоритму стеганографії

```
unit Stegography;

interface

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

{$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 ша 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 хватает свободного места','',$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.
```

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 маркування даних
мікропейментових фотобанків з застосуванням методів стеганографії');
Memo1.Lines.Add('');
Memo1.Lines.Add('Керівник:  Смірнов О.А.');
Memo1.Lines.Add('');
Memo1.Lines.Add('Розробив:  студент   Сосна Олександр Сергійович');
Memo1.Lines.Add('           гр. КБ-20');
Memo1.Lines.Add('');
Memo1.Lines.Add('м. Кропивницький 2024');
Memo1.Lines.Add('');
end;

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