Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2021 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-20МЗ
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Смірнов О.А.
«____» _____ 2021 р.

Керівник проекту
кандидат фізико-математичних наук, доцент
_____ Наталія ЯКИМЕНКО
«____» _____ 2021 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Центр *Заочної та дистанційної освіти*

Кафедра *Кібербезпеки та програмного забезпечення*

Рівень вищої освіти *магістр*

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

Спеціальність *123 "*Комп'ютерна інженерія*"*

Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2021 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Смірнову Олексію Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів*

2. Керівник роботи *Якименко Наталія Миколаївна, канд. фіз.-мат. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 41-13 від 02.08.2021 року

3. Строк подання студентом роботи до захисту *10.12.2021 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів*

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 в промислову експлуатацію
6. Наукова новизна
7. Економічна ефективність розробленої програми.
8. Заходи з охорони праці та техніки безпеки
9. Висновки.

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

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Савеленко Г.В.	05.10.2021	14.11.2021
Охорона праці	Оришака О.В.	06.10.2021	16.11.2021

7. Дата видачі завдання « 6 » вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2021 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2021 р.	
3.	Розробка моделі компонента	20.10.2021 р.	
4.	Розробка структур даних	25.10.2021 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2021 р.	
6.	Програмування алгоритмів	10.11.2021 р.	
7.	Розрахунок економічної ефективності	13.11.2021 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2021 р.	
9.	Оформлення ПЗ	17.11.2021 р.	
10.	Попередній захист роботи	10.12.2021 р.	

Дата видачі завдання
« 6 » вересня 2021 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2021 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Смірнов О.А. Дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2021.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи стеганографічного захисту інформаційних ресурсів.

Метою розробки є дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

Об'єктом дослідження є процес стеганографічного захисту інформаційних ресурсів.

Предметом дослідження є методи стеганографічного захисту інформаційних ресурсів.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

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

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

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows XP/Vista/7/8/10.

Програму розроблено в середовищі Delphi 10.4 Sydney.

Ключові слова: комп'ютерна інженерія, стеганографічний захист, інформаційні ресурси

ABSTRACT

Smirnov O.A. Research and software implementation of the system of steganographic protection of information resources. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2021

In this final qualification work on the second (master's) level of higher education the software which is intended for system of steganographic protection of information resources is developed.

The purpose of development is research and software implementation of the system of steganographic protection of information resources.

The object of research is the process of steganographic protection of information resources.

The subject of research is the methods of steganographic protection of information resources.

Research methods are based on methods of information security theory, methods of mathematical statistics, methods of software development.

The result is the software implementation of the system of steganographic protection of information resources.

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.

Developed user-friendly interface. Instructions for working with software are given.

The program can be used on an IBM PC with Windows XP / Vista / 7/8/10.

The program is developed in the environment of Delphi 10.4 Sydney.

Keywords: computer engineering, steganographic protection, information resources

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

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

ВСТУП

Актуальність теми. У століття високих технологій інформація представляється найбільшою цінністю. Тому не дивно, що останнім часом створюється безліч засобів для її захисту. Серед відповідних напрямків найбільш розвинена криптографія – алгоритми постійно вдосконалюються, доводиться їхня стійкість.

Але в цього напрямку є, щонайменше, два недоліки. По-перше, на відміну від теоретичних принципів, у конкретні програмні реалізації можуть закрадатися помилки, що приводять до розшифровки за час, менший чим розрахунковий. По-друге, очевидно, що у зв'язку з розвитком технологій через якийсь час перебір, що займає на сучасному встаткуванні не один рік або навіть десятиліття, буде виконуватися за розумний час.

Стеганографія використовує принципово інший підхід. Вона приховує не тільки інформацію, але й сам факт її наявності. У цьому випадку в зловмисника не буде практично ніяких зачіпок, щоб догадатися, де вона може перебувати.

Основною метою комп'ютерної стеганографії є приховання файлу повідомлення усередині файлу-контейнера. Крім того, така операція повинна залишитися непоміченою – файл-контейнер зобов'язаний не втрачати функцій, а наявність схованого повідомлення повинно бути максимально складно виявити.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

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

– Огляд існуючих систем стеганографічного захисту інформаційних ресурсів.

– Дослідження системи стеганографічного захисту інформаційних ресурсів.

– Програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

Об'єктом дослідження є процес стеганографічного захисту інформаційних ресурсів.

Предметом дослідження є методи стеганографічного захисту інформаційних ресурсів.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод стеганографічного захисту інформаційних ресурсів.

– Розроблено вітчизняний продукт стеганографічного захисту інформаційних ресурсів, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі стеганографічного захисту інформаційних ресурсів.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LV Науково-технічна конференція здобувачів вищої освіти «Наука – виробництву», 2021, основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №12.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Стеганографія – (греч. steganos – таємниця, секрет; graphy – запис, тобто тайнопис) набір засобів і методів приховання факту передачі повідомлення.

У цей час розвиваються методи комп'ютерної стеганографії – самостійного наукового напрямку інформаційної безпеки, що вивчає проблеми створення компонентів приховуваної інформації у відкритому інформаційному середовищі, що може бути сформована обчислювальними системами й мережами.

Сучасне застосування стеганографії:

1) вбудовування інформації з метою сховати незаконну передачу;

2) захист інформації від несанкціонованого доступу;

3) вбудовування цифрових водяних знаків для захисту авторських прав (watermarking);

4) вбудовування ідентифікаційних номерів (fingerprinting);

5) вбудовування заголовків (captioning).

6) камуфляж програмного забезпечення.

Цифрові водяні знаки (ЦВЗ) застосовуються для захисту від копіювання й несанкціонованого використання мультимедійної інформації й полягає у вбудовуванні в захищаємий об’єкт, невидимих міток – ЦВЗ. Найбільш підходящими об'єктами захисту за допомогою ЦВЗ є нерухливі зображення, файли аудио й відеоданих.

Нерідко методи стеганографії використовують для камуфлювання програмного забезпечення. У тих випадках, коли використання програм незареєстрованними користувачами є небажаним, воно може бути

закамуфльоване під стандартні універсальні програмні продукти (наприклад, текстові редактори) або приховано у файлах мультимедіа.

У сучасній стеганографії, у цілому, можна виділити в напрямки: технологічну стеганографію й інформаційну стеганографію.

До методів технологічної стеганографії ставляться методи, які засновані на використанні хімічних або фізичних властивостей різних матеріальних носіїв інформації.

Хімічні методи стеганографії зводиться майже винятково до застосування невидимого чорнила, до яких ставляться органічні рідини й симпатичні хімікалії.

До фізичних методів можна віднести мікрокрапки, різного виду схованки й методи камуфляжу. У цей час фізичні методи становлять інтерес в області дослідження різний стандартних носіїв інформації засобів обчислювальної, аудіо й відео техніки. Крім цього, з'явився цілий ряд нових технологій, які, базуючись на традиційної стеганографії, використовують останні досягнення мікроелектроніки (голограми, кінеграми).

До інформаційної стеганографії можна віднести методи лінгвістичної й комп'ютерної стеганографії.

Лінгвістичні методи стеганографії підрозділяються на дві основні категорії: умовний лист і семаграми.

Існують три види умовного листа: жаргонний код, пустишечний шифр і геометрична система.

Жаргонний код – коли підміняється сенс слова. Виглядає як звичайне, а на жаргоні значить зовсім інше.

Пустишечний шифр – у тексті мають значення лише деякі певні букви або слова.

Геометрична форма – при її застосуванні слова, що мають значення, розташовуються на сторінці в певних місцях або в крапках перетинання геометричної фігури заданого розміру.

Семаграми – таємні повідомлення, у яких шифропозаченням є будь-які символи, крім букв і цифр. Ці повідомлення можуть бути передані, наприклад, у рисунку, що містить крапки й тирі для читання по коду Морзе.

Комп'ютерна стеганографія – сукупність стеганографічних методів, які реалізуються на основі комп'ютерної техніки й програмного забезпечення в рамках окремих обчислювальних або керуючих систем, корпоративних або глобальних обчислювальних мереж.

При використанні методів комп'ютерної стеганографії повинні враховуватися наступні умови:

– супротивник може мати повне подання про стеганографічну систему й деталі її реалізації. Єдиною інформацією, що повинна залишатися йому невідомою, – це ключ, за допомогою якого можна встановити факт присутності схованого повідомлення і його зміст;

– якщо супротивникові якимось образом удалося довідатися про факт існування схованого повідомлення, то це не повинне дозволити йому витягти подібні повідомлення з інших стеганограм до того часу, поки ключ утаємничений;

– треба щоб зловмисник був позбавлений яких-небудь технічних і інших переваг у розпізнаванні або розкритті змісту таємних повідомлень.

Нижче буде обговорено основні теоретичні положення комп'ютерної стеганографії й розглянуті деякі методи приховання даних в інформаційному середовищі, що може бути підтримана обчислювальними системами й мережами.

За аналогією із криптографічними системами, у стеганографії розрізняють системи із секретним ключем і системи з відкритим ключем.

У стеганографічній системі із секретним ключем використовується один ключ, що повинен бути заздалегідь відомий абонентам до початку схованого обміну секретними повідомленнями або пересланий по захищеному каналу.

У стегосистемі з відкритим ключем для вбудовування й добування таємного повідомлення використовуються різні ключі, причому вивести один

ключ із іншого за допомогою обчислень неможливо. Один із ключів (відкритий) може передаватися вільно по незахищеному каналу зв'язку, а другий, секретний ключ, – по захищеному каналу. Дана схема добре працює при взаємній недовірі відправника й одержувача.

З огляду на все різноманіття стеганографічних систем, зведемо їх до наступних типів: безключових стегосистем**,** систем із секретним ключем, систем з відкритим ключем і змішаних стегосистем.

Безключові стегосистем

Для функціонування безключових стегосистем не потрібно ніяких додаткових даних у вигляді стегоключа крім алгоритму стеганографічного перетворення.

Безпека безключових стегосистем заснована на таємності використовуваних стеганографічних перетворень. Це суперечить основному принципу Керкхоффса для систем захисту інформації. Найчастіше для підвищення безпеки безключових систем, перед початком процесу стеганографічного приховання попередньо виконується шифрування приховуваної інформації. Ясно, що такий підхід збільшує захищеність усього процесу зв'язку, оскільки це ускладнює виявлення схованого повідомлення. Однак, “сильні” стеганографічні системи, як правило, не мають потреби в попередньому шифруванні приховуваних повідомлень.

Стегосистеми із секретним ключем

Дотримуючись закону Керкхоффса, безпека системи повинна ґрунтуватися на деякій секретній інформації, без знання якої не можна витягти з контейнера секретну інформацію. У стегосистемах така інформація називається ***стегоключем***. Відправник, вбудовуючи секретне повідомлення в обраний контейнер **C**, використовує секретний стегоключ k. Якщо використовуваний у стеганографіческом перетворенні ключ k відомий одержувачеві, то він зможе витягти сховане повідомлення з контейнера. Без знання такого ключа будь-який інший користувач цього зробити не зможе.

1.2 Область застосування

Основними положеннями сучасної комп'ютерної стеганографії є наступні:

1. Методи приховання повинні забезпечувати автентичність і цілісність файлу.

2. Передбачається, що супротивникові повністю відомі можливі стеганографічні методи.

3. Безпека методів ґрунтується на збереженні стеганографічними перетворенням основних властивостей відкрито переданого файлу при внесенні в нього секретного повідомлення й деякої невідомої супротивникові інформації – ключа.

4. Навіть якщо факт приховання повідомлення став відомий супротивникові через спільника, добування самого секретного повідомлення представляє складне обчислювальне завдання.

У зв'язку зі зростанням ролі глобальних комп'ютерних мереж стає усе більше важливим значення стеганографії.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

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

Розглянемо існуючі програмні продукти, які аналогічні тому, що розробляється у системі.

Quickstego

Quickstego дозволяє приховувати текст у знімках, щоб тільки інші користувачі Quickstego могли витягати й читати сховані секретні повідомлення.

Рисунок 2.1 – Інтерфейс користувача Quickstego

Після того, як текст буде схований у зображенні, збережене зображення як і раніше є «зображенням», воно буде завантажуватися так само, як і будь-яке інше зображення.

Зображення можна зберегти, відправити по електронній пошті, завантажити в Інтернет, як і колись, єдина відмінність полягає в тому, що воно містить схований текст.

Xiao Steganography

Інструмент використовується для приховання текстового повідомлення усередині зображення. Ви можете також сховати зображення усередині іншого зображення.

Рисунок 2.2 – Інтерфейс користувача Xiao Steganography

Openstego

Використовуючи це програмне забезпечення, ви можете або сховати дані (файл) усередині зображення, або витягти дані із зображення.

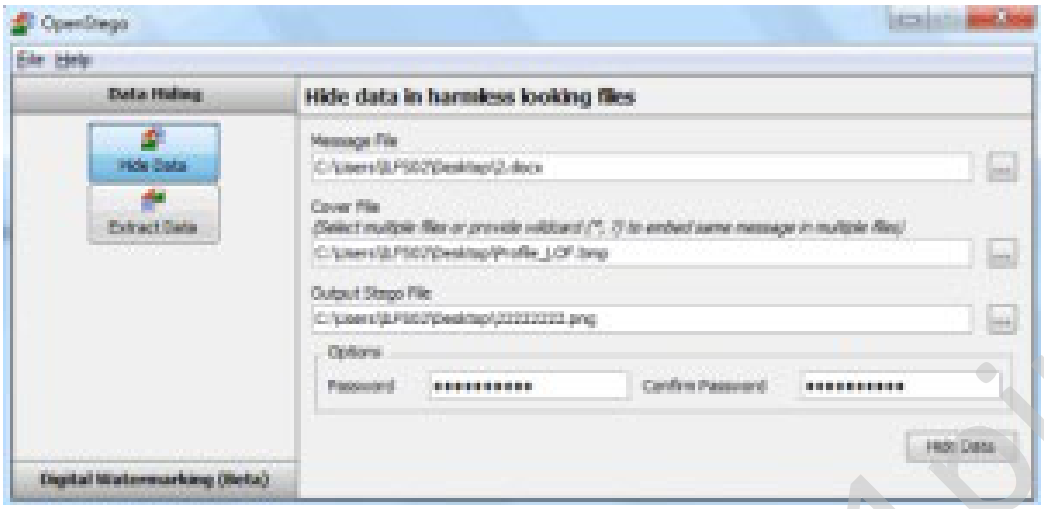

Рисунок 2.3 – Інтерфейс користувача Openstego

Camouflage

Ви можете сховати секретний Txt-файл у стандартному Jpg-зображенні.

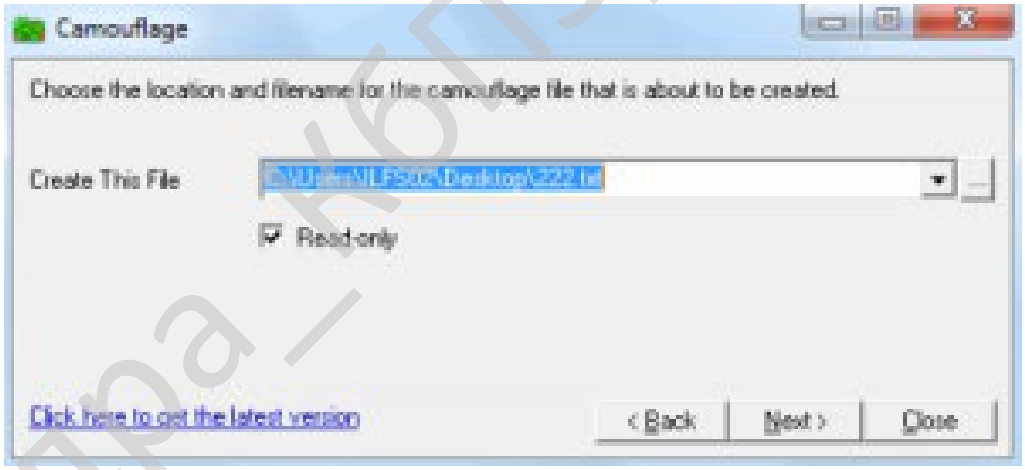

Рисунок 2.4 – Інтерфейс користувача Camouflage

Silenteye

Silenteye – проста у використанні програма крос-платформної стеганографії, яка дозволяє приховувати конфіденційне повідомлення в зображенні або в аудіофайлі.

Рисунок 2.5 – Інтерфейс користувача Silenteye

Steghide

Рисунок 2.6 – Інтерфейс користувача Steghide

Our Secret

Our secret – безкоштовне й простої у використанні програмне забезпечення для стеганографії, яке дозволяє приховувати секретні дані у файлах зображень.

Рисунок 2.7 – Інтерфейс користувача Our Secret

Image Steganography

Image Steganography – безкоштовне програмне забезпечення стеганографії для приховання чутливого тексту або файлів усередині файлів зображень.

Ви можете легко сховати текст або файли різних типів усередині файлів зображень. Він також показує ємність файлу зображення контейнера.

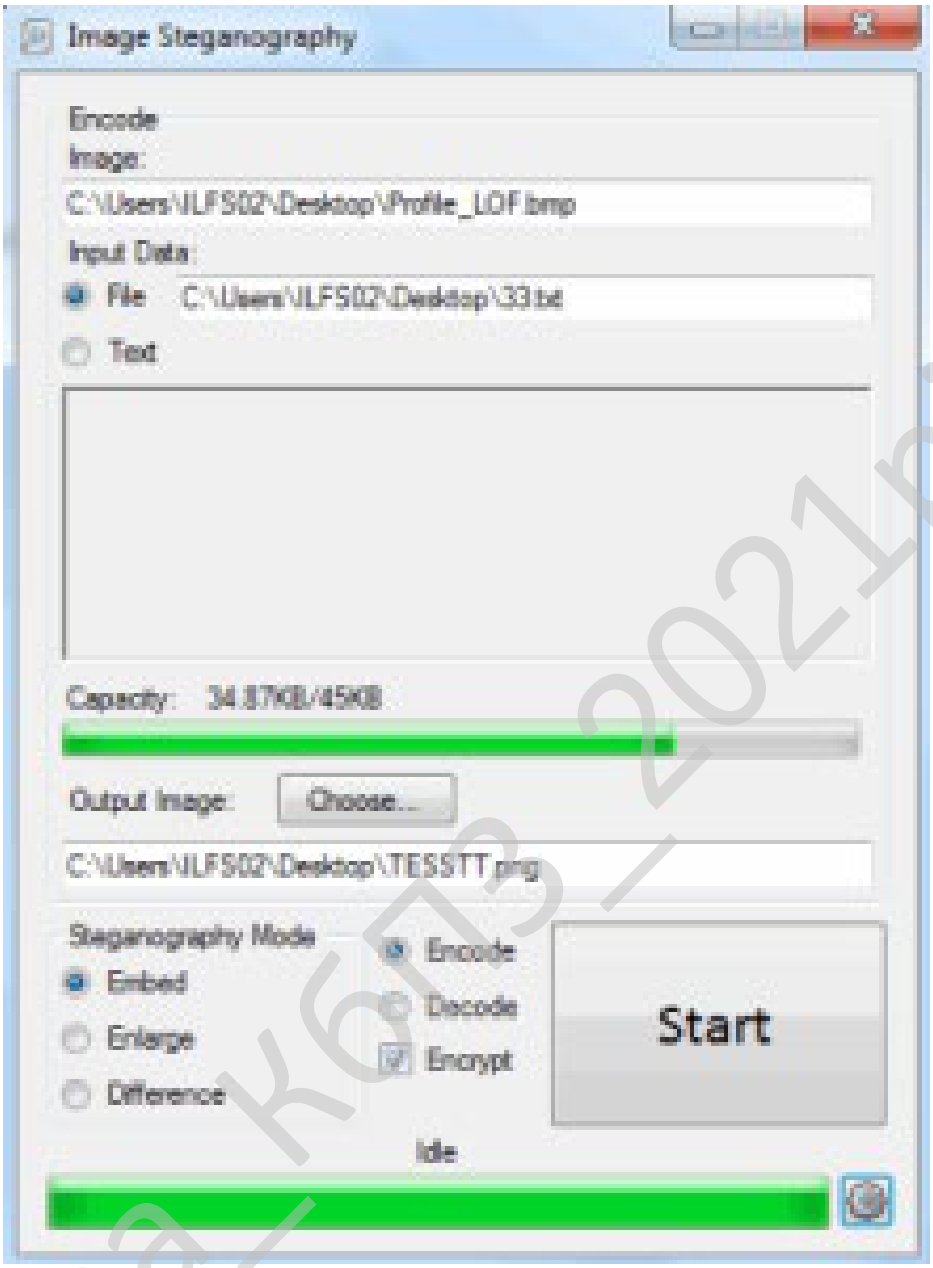

Рисунок 2.8 – Інтерфейс користувача Image Steganography

Steganofile

Steganofile дозволяє приховувати ваш секретний файл в одному або декількох файлах хоста.

Він має кнопки Encode і Decode на своєму інтерфейсі.

Ви можете сховати свій файл в іншому файлі (-ах) хоста.

Виберіть папку призначення й укажіть пароль по вашому вибору.

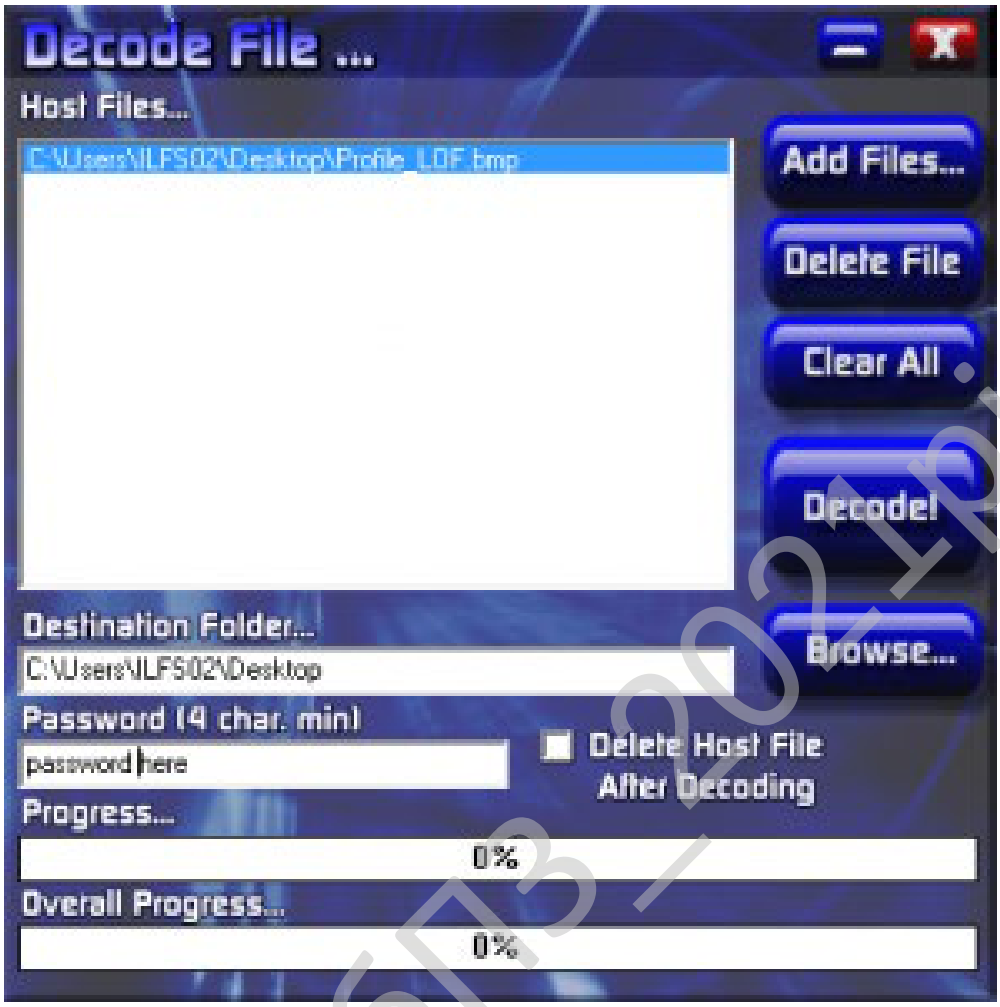

Рисунок 2.9 – Інтерфейс користувача Steganofile

2.2 Обґрунтування вибору засобів для побудови системи 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 Offic e, 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, libSIM Dpp 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 другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи стеганографічного захисту інформаційних ресурсів.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) 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 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 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

У даній роботі ми будемо використовувати наступні визначення:

– Повідомлення – впроваджуване потайливим образом послання, яке необхідно сховати;

– Контейнер (стегоконтейнер) – будь-який об'єкт, використовуваний для таємного впровадження повідомлення;

– Стегосистема – методи й засобу, використовувані для створення схованого каналу для передачі інформації;

– Стегоканал – канал для передачі стегоконтейнера;

– Ключ – ключ для одержання схованого змісту з контейнера (використовується не завжди).

Протягом усього XX століття активно розвивалася як стеганографія, так і наука про визначення факту впровадженої інформації в контейнер – стегоанализ (за сенсом – атаки на стегосистему). Але сьогодні ми спостерігаємо новий і небезпечний тренд: усе більше й більше розроблювачів шкідливого ПЗ й засобів кібершпигунства прибігає до використання стеганографії. Більшість антивірусних розв'язків на сьогоднішній день не захищають від стеганографії або захищають слабко, меж тем, потрібно розуміти, що кожний заповнений контейнер небезпечний. У ньому можуть бути сховані дані, які ексфільтруються шпигунським ПЗ, або комунікація шкідливого ПЗ з командним центром, або нові модулі шкідливого ПЗ.

На сьогодні вченими розроблені й випробувані різні алгоритми й методи стеганографії, ми відзначимо наступні:

– LSB-стеганографія (повідомлення ховається в молодших бітах (можливе використання одного або декількох молодших біт) контейнера. Чим менше біт задіяне, тем менше артефактів одержує оригінальний контейнер після впровадження.

– Метод, заснований на прихованні даних у коефіцієнтах дискретного косинусного перетворення (далі ДКП) – різновид попереднього методу, яка активно використовується, наприклад, при впровадженні повідомлення в контейнер формату JPEG. При інших рівних, такий контейнер має трохи меншу ємність чому в попередньому методі, у тому числі за рахунок того, що коефіцієнти «0» і «1» залишаються незмінними – впровадження повідомлення в них неможливо.

– Метод приховання інформації за допомогою молодших біт палітри – цей метод за сенсом є варіантом загального методу LSB, але інформація вбудовується не в найменш значущі біти контейнера, а в найменш значущі біти палітри, очевидний недолік такого методу – низька ємність контейнера.

– Метод приховання інформації в службових полях формату – досить простий метод, заснований на використанні службових полів заголовка контейнера для зберігання повідомлення. Очевидні недоліки – низька ємність контейнера й можливість виявлення впроваджених даних за допомогою звичайних програм для перегляду зображення (які іноді дозволяють бачити вміст службових полів).

– Метод вбудовування повідомлення – полягає в тому, що повідомлення вбудовується в контейнер, потім за допомогою схеми, відомої обом сторонам, видобувається. Можна вмонтувати кілька повідомлень в один контейнер, за умови, що способи їх впровадження ортогональні.

Широкосмугові методи, які підрозділяються на:

– метод псевдовипадкової послідовності; використовується секретний сигнал, який моделюється псевдовипадковим сигналом;

– метод стрибучих частот: частота несучого сигналу міняється по певному псевдовипадковому закону.

Метод оверлея – за сенсом не є справжньої стеганографією, заснований на тому, що деякі формати містять у заголовку розмір даних, або ж оброблювач цих форматів буде читати файл до маркера кінця даних. Прикладом такого методу є добре відомий метод «RAR-jpeg», який заснований на конкатенації графічного файлу у форматі JREG і RAR -архіву. ПЗ для перегляду JPEG буде зчитувати інформацію до границі, зазначеної в заголовку файлу, а RAR-архіватор відкине все, що перебуває до сигнатури «RAR!», яка позначає початок архіву. Таким чином, якщо такий файл відкрити в переглядачі графічних файлів – ми побачимо картинку, а якщо в RAR-архіваторі – уміст RAR-архіву. Очевидні недоліки такого підходу полягають у тому, що оверлей, доданий до контейнера, легко виділяємо при візуальному дослідженні такого файлу.

У цій роботі ми розглядаємо тільки методи приховання інформації в графічних контейнерах і в мережних пакетах, але область застосування стеганографії значно ширше.

За недавній час ми спостерігали використання стеганографії в наступних шкідливих програмах і засобах кібершпигунства:

– Microcin (AKA six little monkeys);

– Nettraveler;

– Zberp;

– Enfal (its new loader called Zero.T);

– Shamoon;

– Kins;

– Zeusvm;

– Triton (Fibbit).

Чому автори шкідливого ПЗ все активніше використовують стеганографію у своїх розробках? Ми бачимо три головні причини:

– Це дозволяє їм сховати сам факт завантаження/вивантаження даних, а не тільки самі дані.

– Допомагає обійти DPI-системи, що актуально в корпоративних мережах.

– Використання стеганографії може дозволити обійти перевірку в AntiAPT-Продуктах, оскільки останні не можуть обробляти всі графічні файли (їх занадто багато в корпоративних мережах, а алгоритми аналізу досить дорогі).

Для кінцевого користувача детектування стегоконтейнера може бути нетривіальним завданням.

Використання стеганографії сьогодні – дуже популярна ідея серед авторів шкідливого й шпигунського ПЗ.

Антивірусні засоби взагалі й засобу захисту периметра зокрема мало що можуть зробити із заповненими контейнерами: їх дуже важко виявити оскільки вони виглядають як звичайні графічні (і не тільки) файли.

Усі існуючі програми для детектування стеганографії за сенсом є PoC – Proof-of-Concept, і їхня логіка не може бути імплементована в промислові засоби захисту через низьку швидкість роботи, не занадто високого рівня детектування, і іноді навіть іноді – через помилки в математику (ми бачили й такі випадки).

Вище був наведений список шкідливого ПЗ (втім, далеко не повний), яке використовує стеганографію для приховання своєї комунікації. Розглянемо конкретний приклад із цього списку – шкідливий завантажник Zero.T.

Цей завантажник був виявлений наприкінці 2016 року, але перший опис був опублікований колегами з Proofpoint.

Його назвали Zero.T, тому що такий рядок присутня в його коді, що виконується (у шляху до pdb-файлу проекту):

Не зупиняючись тут на влученні в систему й закріпленні в ній, відзначимо, що Zero.T завантажує корисне навантаження у вигляді Bitmap-файлів.

Обробляє їхнім особливим образом, після чого одержує шкідливі модулі.

На перевірку ці три Bmp-файлу виявилися картинками.

Але картинки ці не зовсім звичайні. Це – заповнені контейнери. У кожному з яких трохи (алгоритм допускає варіативність) молодших значущих біт замінені на корисне навантаження.

Тому що ж визначити чи є картинка заповненим контейнером чи ні? Існують різні способи, але найпростіший з них – візуальна атака. Суть його полягає у формуванні нових зображень на основі вихідн, що полягають із найменш значущих біт різних колірних площин.

Розглянемо на прикладі зображення зі світлиною Стива Джобса. Застосуємо до цього зображення візуальну атаку, побудуємо нові зображення з окремих значущих біт відповідних розрядів. На другому й третьому зображенні помітні області з високою ентропією (високою щільністю даних) – це і є впроваджене повідомлення.

Просто, чи не так? І так, і немає. Це просто, тому що аналітик (і навіть простий користувач!) може з легкістю побачити впроваджені дані). І складно тому що такий аналіз досить важко автоматизувати. На щастя, учені вже давно розробили кілька методів виявлення заповнених контейнерів, заснованих на статистичних характеристиках зображення. Але всі вони ґрунтуються на припущенні, що впроваджене повідомлення має високу ентропію. Найчастіше це дійсно так: оскільки ємність контейнера обмежена, повідомлення перед впровадженням стискується й/або шифрується, тобто його ентропія підвищується.

Але наш приклад з реального життя – шкідливий завантажник Zero.T, – не стискає свої модулі перед впровадженням! Замість цього він збільшує кількість використовуваних найменш значимих біт: 1,2 або 4. Так, використання більшої

кількості найменш значущих біт приводить до появи візуальних артефактів у зображенні, помітних простому користувачеві. Але ж мова йде про автоматичний аналіз. Питання, яке нам має бути з'ясувати: чи придатні статистичні методи для виявлення впроваджених повідомлень у тому випадку, якщо їх ентропія не висока?

Статистичні методи аналізу: гістограмний метод

Описуваний метод, запропонований в 2000 році Андресом Вестфелдом і Андреасом Пфітцманом, також відомий як «хі-квадрат»-метод. Спробуємо викласти його суть.

Увесь растр аналізується, для кожного кольору вважається кількість крапок такого кольору в растрі (для простоти тут говоримо про зображення, що має одну колірну площину). Метод виходить із припущення, що кількість крапок двох сусідніх квітів («сусідні» кольори – кольору, які відрізняються тільки найменш значимим бітом) різниться суттєво для нормального, звичайного зображення (порожнього контейнера). І кількість пікселів таких квітів є приблизно однаковим для заповненого контейнера.

Візуально можна уявляти собі алгоритм таким чином, це досить просто для розуміння.

Говорячи строго, алгоритм полягає в послідовному виконанні наступних кроків.

Теоретично очікувана частота зустрічальності пікселів кольору і після впровадження повідомлення розраховується в такий спосіб.

Обмірювана частота входження символу певного кольору визначена.

Хі-квадрат критерій для кількості ступенів волі k-1 розраховується.

P – це ймовірність того, що розподілу ni і ni* при цих умовах рівні. Вона розраховується за допомогою інтегрування функції гладкості:

Зрозуміло, провели перевірку застосовності цього методу для детектування заповнених стегоконтейнерів.

Граничні значення розподілу хі-квадрат для p=0,95 і p=0,99 відповідно 101.9705929 і 92.88655838. Таким чином, для зон, у яких розраховане значення хі-квадрат менше граничного, можна прийняти вихідну гіпотезу «розподіл частот сусідніх квітів – однакове, отже, це заповнений стегоконтейнер».

Дійсно, якщо подивитися на зображення для візуальної атаки, нескладно помітити, що ці області містять впроваджене повідомлення. Таким чином, для впроваджених повідомлень із високою ентропією метод працює.

Статистичні методи аналізу: RS-Метод

Ще один статистичний метод виявлення заповнених стегоконтейнерів був запропонований Джессикой Фрідріх, Мирославом Гольяном і Андреасом Пфітцманом в 2001 році. Він називається RS-Метод, де RS означає « регулярний-сингулярний».

Усе зображення розділяється на безліч груп пікселів, далі для кожної групи застосовується спеціальна фліппінг-процедура. На підставі значення функції-дескримінанта до й послу застосування фліппінга всі групи діляться на регулярні, сингулярні й невикористовувані.

Алгоритм грунтується на припущенні, що кількість регулярних і сингулярних груп пікселів в оригінальному зображенні й у зображенні після застосування фліппінга повинне бути приблизно рівним. Якщо кількість таких груп суттєво міняється в процесі застосування фліппінга, це значить, що досліджуване зображення є заповненим контейнером.

По кроках алгоритм працює в такий спосіб:

– Зображення розділяється на групи з n pixels (x_1, …, x_n).

– Описується т.зв. дискримінант-функція, яка зіставляє кожній групі пікселів G = (x_1, …, x_n). дійсне число f(x_1, …, x_n) ∈ R.

– Ми можемо визначити дискримінант-функцію для групи пікселів (x_1, …, x_n).

– Також ми визначаємо функцію фліппінга.

– На підставі значень дискримінант-фукнції до й після фліппінга, усі групи пікселів діляться на регулярні, сингулярні й невикористовувані:

Ми провели дослідження й цього методу теж, одержавши наступні результати. Ми використовували ті ж заповнений і порожній контейнери, що й у попередньому досвіді.

Зверніть увагу, що цей метод атаки не виносить бінарного вердикту «чи містить цей контейнер впроваджене повідомлення», замість цього він визначає зразкову довжину впровадженого повідомлення (у відсотках).

З результатів вище зрозуміло, що для порожнього повідомлення цей метод виніс вердикт про заповнення менш 1% контейнера, а для заповненого – про заповнення приблизно 44% контейнера. Очевидно, що результати злегка неточні. Оборотний увага на заповнений контейнер: з візуальної атаки однозначно випливає, що заповнене більш 50% контейнера, у той час як RS -атака говорить нам, що заповнено 44% контейнера. Тому ми можемо застосовувати цей метод якщо встановимо якийсь «поріг спрацьовування»: наші експерименти показали, що 10% є достатнім порогом надійності. Якщо RS-атака затверджує, що більш 10% контейнера заповнене – можна довіряти цьому вердикту й маркірувати контейнер як заповнений.

Тепер прийшов час з'ясувати, придатні чи два розглянуті способи для тестування в реальних умовах на контейнерах Zero.T, повідомлення в яких мають звичайну ентропію.

Ми провели відповідні тести й от результати.

Очевидно, що на зображеннях з низькою ентропією атака типу «хі-квадрат» не застосовна – результати або незадовільні, або не цілком точні, зате RS-атака відробила відмінно: в обох випадках була визначена наявність схованого повідомлення. Але що ж робити якщо автоматичні методи аналізу показали відсутність впровадженого повідомлення, а ми усе ще підозрюємо це?

Можна використовувати конкретні процедури для добування корисного навантаження, розроблені для конкретних сімейств шкідливого ПЗ. Так, для розглянутого в цій роботі завантажника Zero.T ми написали власну процедуру екстракції впровадженого повідомлення, яка схематично працює в такий спосіб.

Очевидно, що, якщо в підсумку ми одержали валідний результат (у цьому випадку – файл, що виконується), вихідне зображення було контейнером.

DNS-tunneling: теж стеганографія?

Чи можливо вважати використання DNS-туннеля підвидом стеганографії Дійсно, так. Для початку давайте згадаємо, як виглядає схема DNS-тунелю загалом.

У закритій мережі з користувацької машини посилає запит на «резолв» домену, наприклад:

wl8nd3Ddincgyaaj7Hh0H56a8nd3Ddincgyalfdhburwzmt[.]imbadguy[.]com

(де доменне ім'я другого рівня не несе значеннєвого навантаження). Локальний DNS-сервер передає запит зовнішньому DNS-серверу. Тому, у свою чергу, невідоме ім'я 3-го рівня й запит передається далі. Таким чином, по ланцюжкові перенапрямків від одного DNS-сервера до іншого запит досягає DNS-сервера домену imbadguy[.]com.

Замість дозволу запиту на сервері зловмисник може витягти з отриманого домену потрібну йому інформацію, розшифрувавши першу частину його імені. Наприклад, таким способом можна передати інформацію про систему користувача. У відповідь DNS-сервер зловмисника так само посилає якусь інформацію в зашифрованому виді, передаючи її в доменному імені 3-го рівня або вище.

Таким чином, зловмисник має про запас для кожного DNS резолва 255 символів, до 63 символів для піддоменів. 63 символу в одну сторону – 63 у відповідь, 63 – туди, 63 – назад… Непоганий канал для передачі даних! А головне – схований, адже неозброєним поглядом не видне, що йде обмін якоюсь додатковою інформацією.

Для фахівців, знайомих з мережними протоколами й зокрема з DNS-туннелюванням, такий дамп трафіка буде виглядати досить підозріло: дуже вуж багато успішних резолвів довгих доменів. У цьому випадку ми спостерігаємо реальний приклад трафіка зловреда Backdoor.Win32.Denis, який використовує DNS тунель як схований канал спілкування зі своїм командним центром.

Виявити DNS-тунель можна за допомогою будь-який відомої IDS, наприклад, Snort, Suiricata або BRO Ids. Як саме розпізнати DNS -туннелювання? Існують різні способи. Наприклад, мабуть, що доменні імена для резолва при туннелюванні значно довше, чим звичайно. У мережі можна знайти достатню кількість варіацій на цю тему:

```
alert udp any any -> any 53 (msg:»Large DNS Query, possible cover channel»; content:»|01 00 00 01 00 00 00 00 00 00|»; depth:10; offset:2; dsize:>40; sid:1235467;)
```

Зовсім примітивно, але зустрічається й таке:

```
Alert udp $HOME_NET and -> any 53 (msg: «Large DNS Query»; dsize: >100; sid:1234567;)
```

Отут можна експериментувати й прагнути знайти свій ідеальний баланс між кількістю неправильних спрацьовувань і знаходженням реального DNS-туннелювання.

На що ще можна звернути увагу крім підозріло великої довжини доменного імені? На аномальний синтаксис доменних імен. Усі ми приблизно представляємо, як виглядають типові домени – це здебільшого букви й цифри. Якщо в доменному імені присутні символи, характерні для Base64 кодування, то це виглядає досить підозріло, чи не так? Якщо при цьому ще й довжина не маленька, то явно варто придивитися більш уважно.

І таких аномалій можна описати досить велика кількість, регулярні вираження нам у допомогу.

Хочеться відзначити, що навіть такий простій по своєму змісту підхід до пошуку DNS-тунелів дає дуже гарні результати. Завдяки декільком таким правилам для IDS ми на вхідному потоці шкідливого ПЗ, виявили декілька нових, невідомих до цього бекдорів, що використовують схований канал спілкування з командним центром – DNS-тунель.

Ми відзначаємо сильний позитивний тренд: усе більше й більше розроблювачів шкідливого ПЗ починає використовувати стеганографію, у тому числі – для приховання комунікації з командним центром і для завантаження модулів. Це дає результат, адже процедури аналізу контейнерів імовірнісні й дорогі, отже, більшість захисних розв'язків не можуть собі дозволити обробляти всі об'єкти, які потенційно можуть бути заповненими контейнерами.

Однак розв'язку є, вони засновані на комбінуванні різних способів аналізу, високошвидкісних предетектах, дослідженні метаданих потенційно заповненого контейнера й т.п.

Такі рішення зараз реалізуються в нашій платформі захисту від таргетованих атак – КАТА. Її використання дозволяє співробітникові служби інформаційної безпеки вчасно довідатися про можливу таргетованій атаці на, що захищається периметр і/або ексфільтрації даних з нього.

3.2 Розробка структурної схеми

В результаті виконання даного магістерської роботи було розроблено програмне забезпечення стеганографічного захисту інформаційних ресурсів. Дана система повинна захищати від несанкціонованого доступу конфіденційну інформацію, що передається по Інтернету та локальних мережах.

Структурна схема розробленого програмного забезпечення представлена на рисунку 3.1.

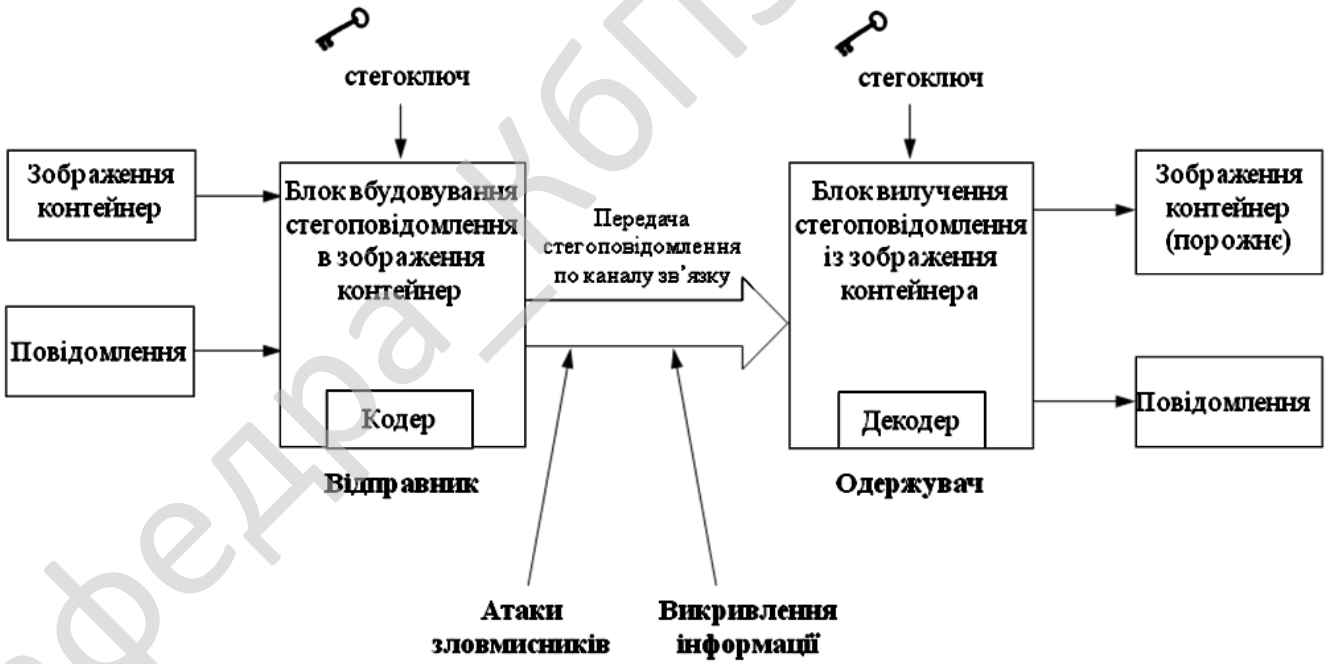

Рисунок 3.1 – Структурна схема системи

В якості даних, що вбудовуються може використовуватися будь-яка інформація: текст, повідомлення, невелике зображення тощо.

Роль контейнеру буде відігравати будь-яке кольорове цифрове зображення, що задовольняє стандартним вимогам до контейнерів для стегоповідомлень.

Стегоключ – секретний ключ, необхідний для приховування інформації. Залежно від кількості рівнів захисту (наприклад, вбудовування заздалегідь зашифрованого повідомлення) в стегосистемі може бути один або декілька стегоключів.

Вбудовування повідомлення в зображення контейнер відбувається за допомогою стегокодера, який крім приховування інформації здійснює також і перешкодостійке кодування.

Після цього зображення з прихованим повідомленням передається по каналу зв'язку, де може зазнавати атак зловмисників, а також викривлень інформації в наслідок перешкод у каналі зв'язку або застосувань алгоритмів стиснення з втратами.

Вилучення повідомлення із зображення контейнера здійснюється за допомогою стегодетектора. Стегодекодер перевіряє наявність прихованого повідомлення і в разі його існування, вилучає інформацію.

3.3 Розробка функціональної схеми

Більш докладну взаємодію між структурними блоками системи представлено на функціональній схемі роботи системи (рисунок 3.2).

Система включає дві процедури:

– процедуру вбудовування конфіденційної інформації в зображення контейнер;

– процедуру вилучення конфіденційної інформації із зображення контейнера.

Робота системи відбувається наступним чином.

Спершу опишемо процедуру вбудовування конфіденційної інформації в зображення контейнер.

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

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

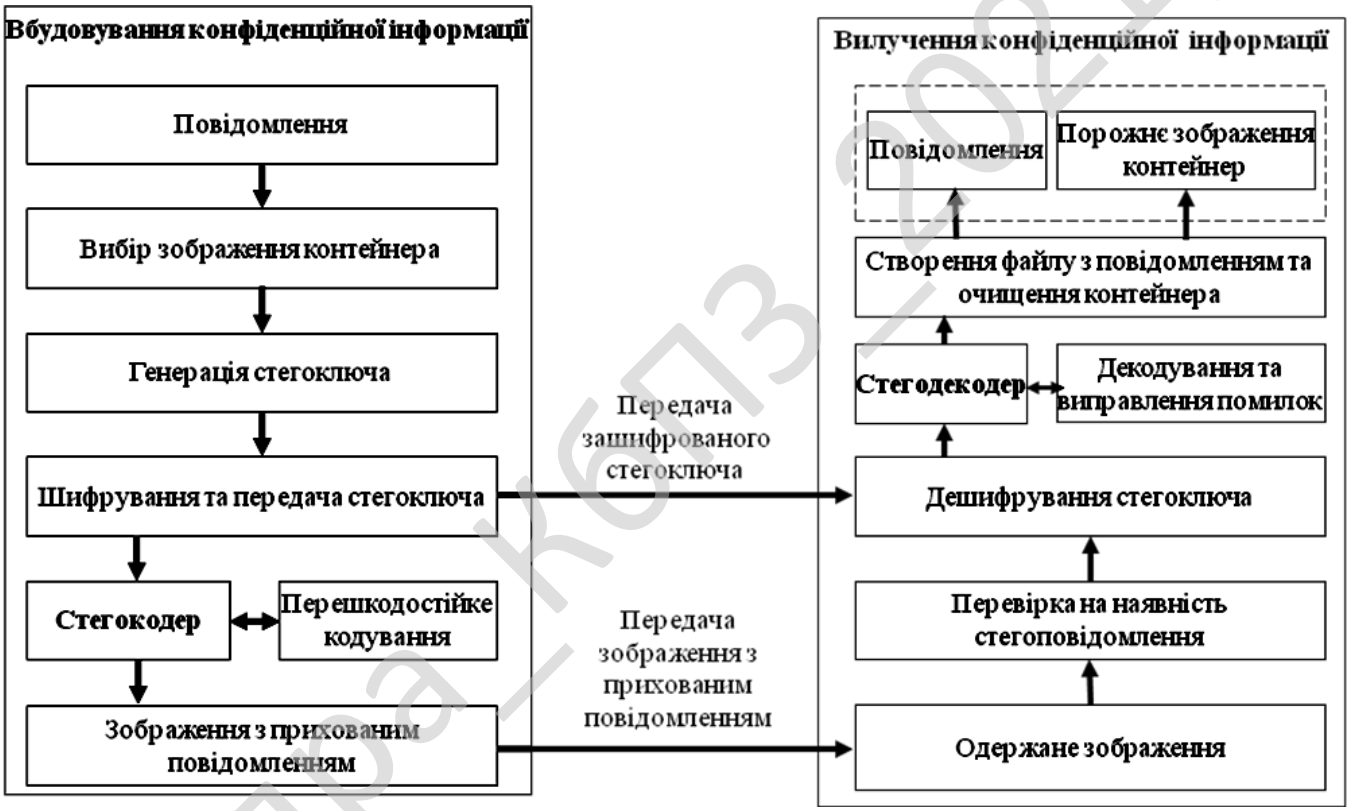

Рисунок 3.2 – Функціональна схема системи

Принцип кодування наступний:

1. Обчислюється контрольна сума пароля.

2. Обчислюється контрольний добуток пароля.

3. Всі закодовані дані представляються як масив байтів.

4. Від кожного байта даних віднімається байт контрольної суми пароля.

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

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

7. Як тільки рядок пароля закінчується знову пер лунадимо на його початок.

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

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

Для більш високої надійності передачі даних, повідомлення кодується за допомогою перешкодостійкого кодека Хеммінга.

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

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

Це відбувається наступним чином.

Спершу отримане повідомлення перевіряють на наявність помилок, за рахунок застосування кодеку Хеммінга.

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

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

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

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

DES є класичною мережею Фейштеля із двома гілками. Дані шифруються 64-бітними блоками, використовуючи 56-бітний ключ. Алгоритм перетворить за кілька раундів 64-бітний вхід в 64-бітний вихід. Довжина ключа дорівнює 56 бітам. Процес шифрування складається із чотирьох етапів.

На першому з них виконується початкова перестановка (IP) 64-бітного вихідного тексту (забілювання), під час якої біти з у відповідності зі стандартною таблицею.

Наступний етап складається з 16 раундів однієї й тої ж функції, що використовує операції зрушення й підстановки.

На третьому етапі ліва й права полови ни виходу останньої (16-й) ітерації міняються місцями.

Нарешті, на четвертому етапі виконується перестановка IP^{-1} результату, отриманого на третьому етапі. Перестановка IP^{-1} інверсна початковій перестановці.

Опишемо спосіб, яким використовується 56-бітний ключ. Спочатку ключ подається на вхід функції перестановки. Потім для кожного з 16 раундів підключ K_i є комбінацією лівого циклічного зрушення й перестановки. Функція перестановки та сама для кожного раунду, але підключи K_i для кожного раунду виходять різні внаслідок повторюваного зрушення біт ключа.

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

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання магістерської роботи, наведена на рисунку 3.3.

Процеси, які є в системі взаємодіють наступним чином.

Процес початку роботи програмного продукту взаємодіє з наступними процесами:

– Процесом введення повідомлення.

– Процесом одержання зображення.

Процес введення повідомлення взаємодіє з наступними процесами:

– Процес вибору зображення контейнера.

– Процесом генерації стегоключа.

Процес генерації стегоключа взаємодіє з наступними процесами:

– Процесом шифрування ключа.

– Процесом передачі ключа.

– Процесом вбудовування стегоповідомлення.

Процес вбудовування стегоповідомлення взаємодіє з наступними процесами:

– Процесом реалізації перешкодостійкого кодування.

– Процесом передачі стегоповідомлення.

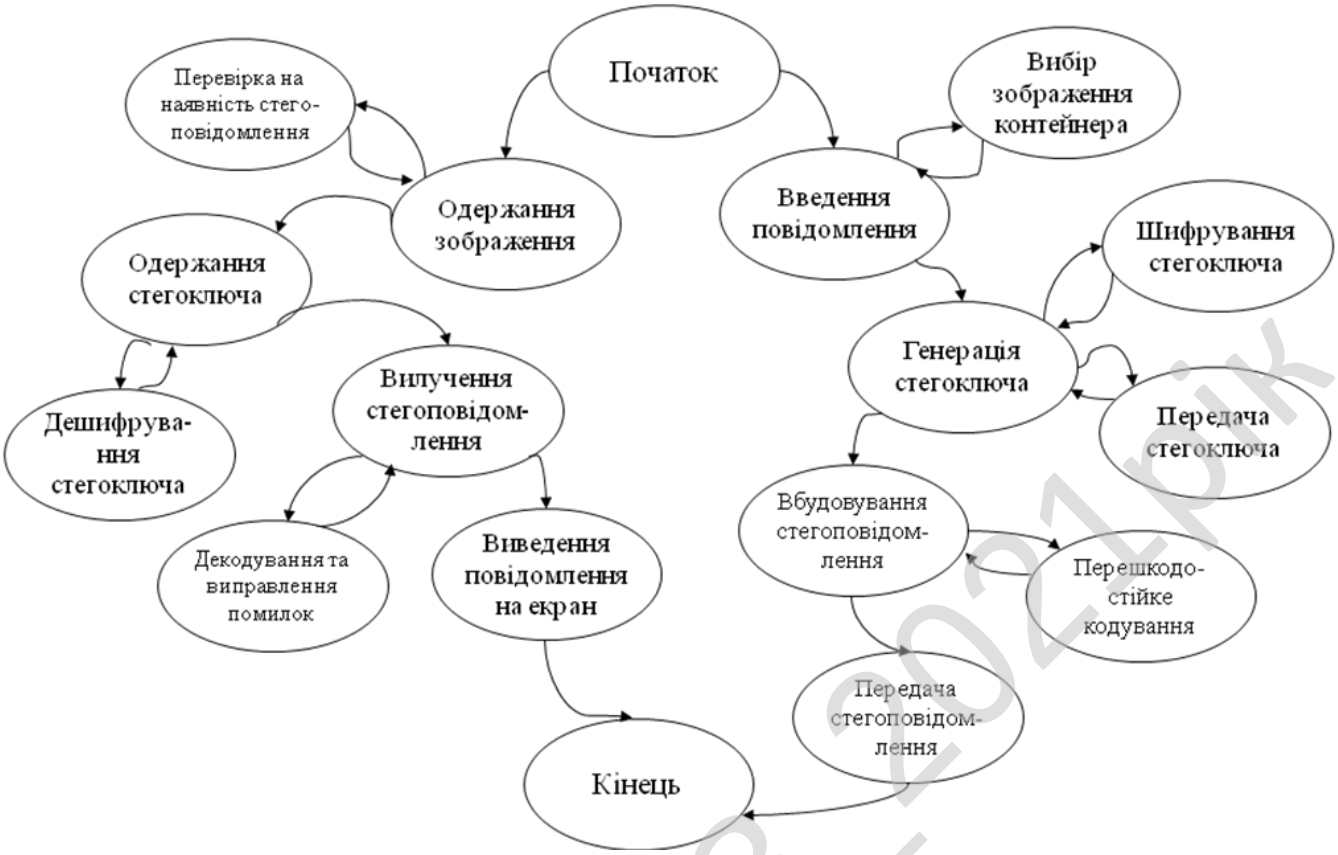

Рисунок 3.3 – Діаграма процесів

З іншого боку, процес одержання зображення взаємодіє з наступними процесами:

– Процесом перевірки на наявність стегоповідомлення.

– Процесом одержання стегоключа.

Останній процес взаємодіє з наступними процесами, які відбуваються у системі:

– Процес дешифрування стегоключа.

– Процес вилучення стегоповідомлення.

Процес вилучення стегоповідомлення взаємодіє з наступними процесами:

– Процесом декодування та виправлення помилок.

– Процесом виведення повідомлення, яке було сховане у зображені, на екран.

Після виконання усіх вищеперерахованих процесів, система закінчує свою роботу.

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

4.1 Блок–схеми та опис алгоритмів функціонування системи

На рисунку 4.1 наведено блок-схему основної програми. Її робота складається з виконання наступних кроків.

Спершу відбувається виведення основного вікна програми. Після цього запускається вікно запиту, чи є необхідним створення стегоповідомлення.

Якщо є необхідність то виконуються наступні ітерації:

– Вводиться повідомлення.

– Відбувається вибір зображення, яке буде контейнером для прихованого повідомлення.

Після цього визначається чи є стегоключ.

Якщо його немає то виконуються наступні дії:

– Відбувається генерація стегоключа.

– Згенерований стегоключ шифрується за алгоритмом DES, та передається одержувачу.

– Повідомлення кодується перешкодостійким алгоритмом Хеммінга.

– Виконується підпрограма вбудовування повідомлення у зображення.

– Повідомлення передається по каналам зв’язку до отримувача.

На приймальній стороні визначають чи потрібно читати стегоповідомлення.

Якщо його потрібно читати, тоді виконуються наступні кроки:

– Відкривається зображення.

– Отримується стегоключ.

– Запускається підпрограма вилучення стегоповідомлення.

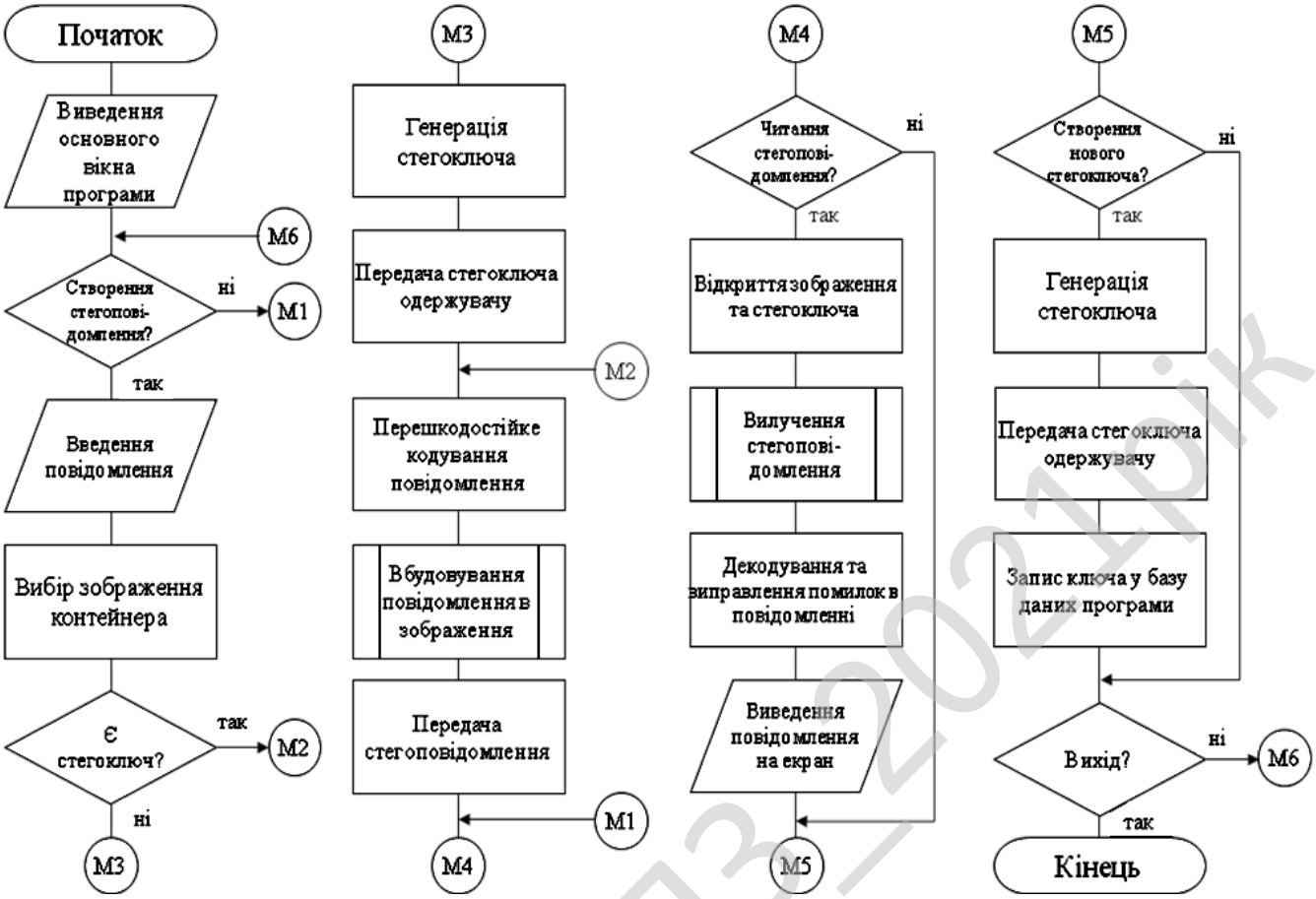

Рисунок 4.1 – Блок-схема роботи основної програми

– Відбувається декодування та виправлення помилок в повідомленні.

– Отримане повідомлення виводиться на екран.

Після цього визначають, чи потрібно створення нового стегоключа.

Якщо потреба у цьому є, тоді виконуються наступні дії:

– Відбувається генерація стегоключа.

– Стегоключ передається одержувачу.

– Згенерований стегоключ записується у базу даних програми.

Так, як у програмі використовується перешкодостійке кодування Хеммінга, наведемо його опис.

Коди Хеммінга є кодами, що самоконтролюються, тобто кодами, що дозволяють автоматично виявляти найбільш імовірні помилки при передачі даних. Для їхньої побудови досить приписати до кожного слова один додатковий

(контрольний) двійковий розряд і вибрати цифру цього розряду так, щоб загальна кількість одиниць у зображенні будь-якого числа була, наприклад, парнею. Одиночна помилка в якому-небудь розряді переданого слова (у тому числі, може бути, і в контрольному розряді) змінить парність загальної кількості одиниць. Лічильники за модулем 2, що підрах овують кількість одиниць, які втримуються серед двійкових цифр числа, можуть давати сигнал про наявність помилок.

При цьому, зрозуміло, ми не одержуємо ніяких вказівок про те, у якому саме розряді відбулася помилка, і, отже, не маємо можливості виправити неї. Залишаються непоміченими також помилки, що виникають одночасно у двох, у чотирьох або взагалі в парній кількості розрядів. Втім, подвійні, а тим більше чотириразові помилки покладаються малоймовірними.

Наведемо частину коду програми, яка реалізує код Хеммінга.

```
//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;
```

На рисунку 4.2 зображено блок-схему роботи підпрограми створення стегоповідомлення.

З рисунку бачимо, що підпрограмма працює наступним чином.

Спершу відбувається відкриття стегоключа, повідомлення та зображення.

Після цього відбувається розбиття зображення контейнера на фрагменти.

Наступним кроком є створення та обнуління лічильників фрагментів.

Після цього вибирається пік сель поточного фрагменту відповідно до стегоключа.

Відповідно до обраного пікселя відбувається розрахунок значення яскравості цього пікселя.

Якщо у піксель потрібно записати 0, тоді збільшуємо яскравість пікселя.

У іншому випадку зменшуємо яскравість пікселя, тобто записуємо у нього 1.

Збільшуємо лічильник фрагментів.

Поки не оброблений останній фрагмент, виконуємо усі вище перераховані операції над інформацією та пікселями.

На рисунку 4.3 зображена блок-схема роботи підпрограми читання стегоповідомлення. Вона працює наступним чином.

Спершу відбувається відкриття стегоключа та зображення.

Після цього відбувається розбиття зображення контейнера на фрагменти.

Створюються та обтулюються лічильники фрагментів.

Відбувається вибір пікселя з поточного фрагменту відповідно до стегоключа.

Розраховується середнє значення яскравості інших пікселів у фрагменті.

Якщо обраний піксель є більш яскравим, то робиться висновок, що у піксель записано 1.

У іншому випадку, вважається, що у піксель записано 0.

Відбувається збільшення лічильника фрагментів.

Вищеперераховані операції відбуваються доти, доки не оброблено останній фрагмент.

На цьому підпрограма закінчує свою роботу.

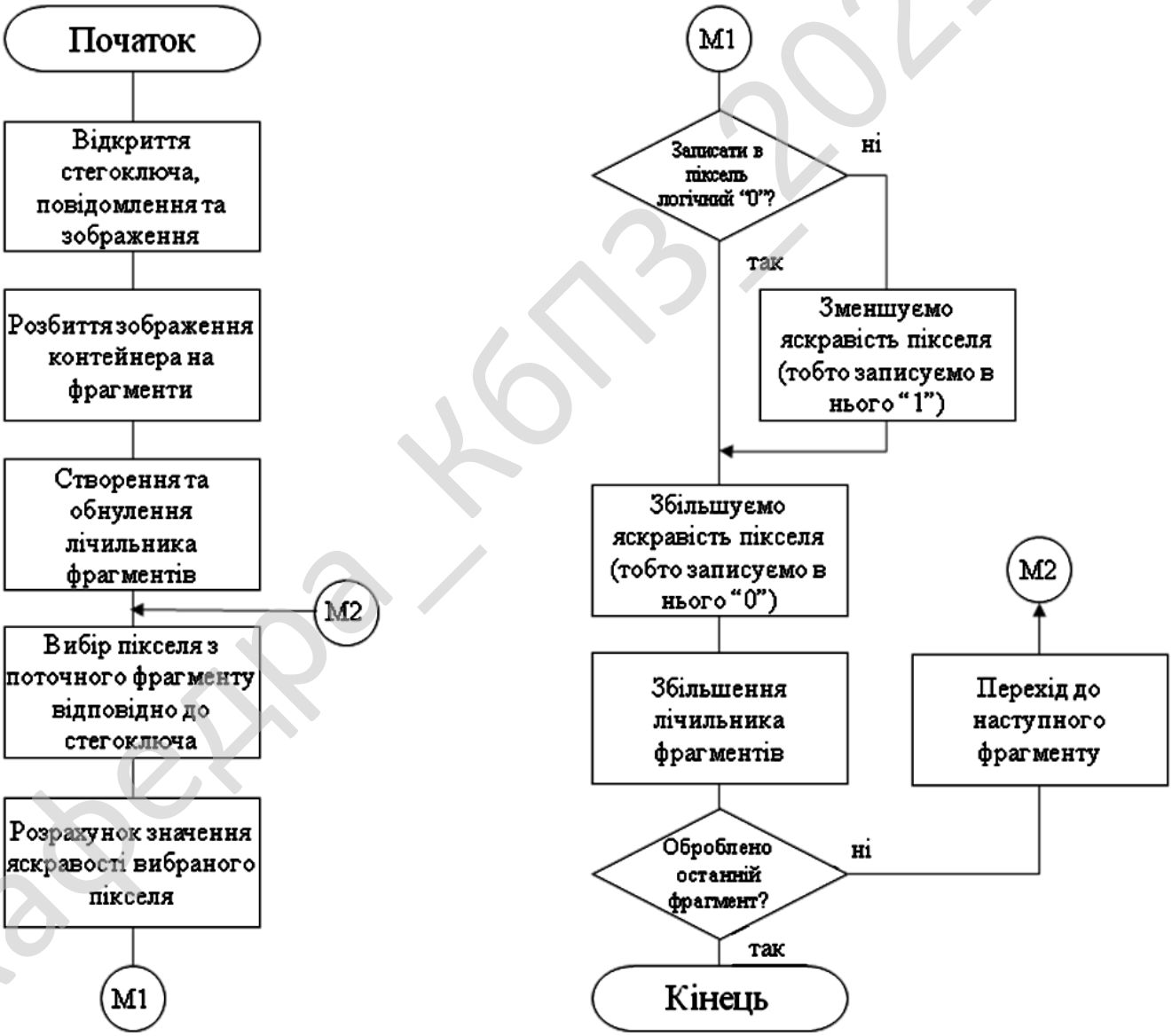

Рисунок 4.2 – Блок-схема роботи підпрограми створення стегоповідомлення

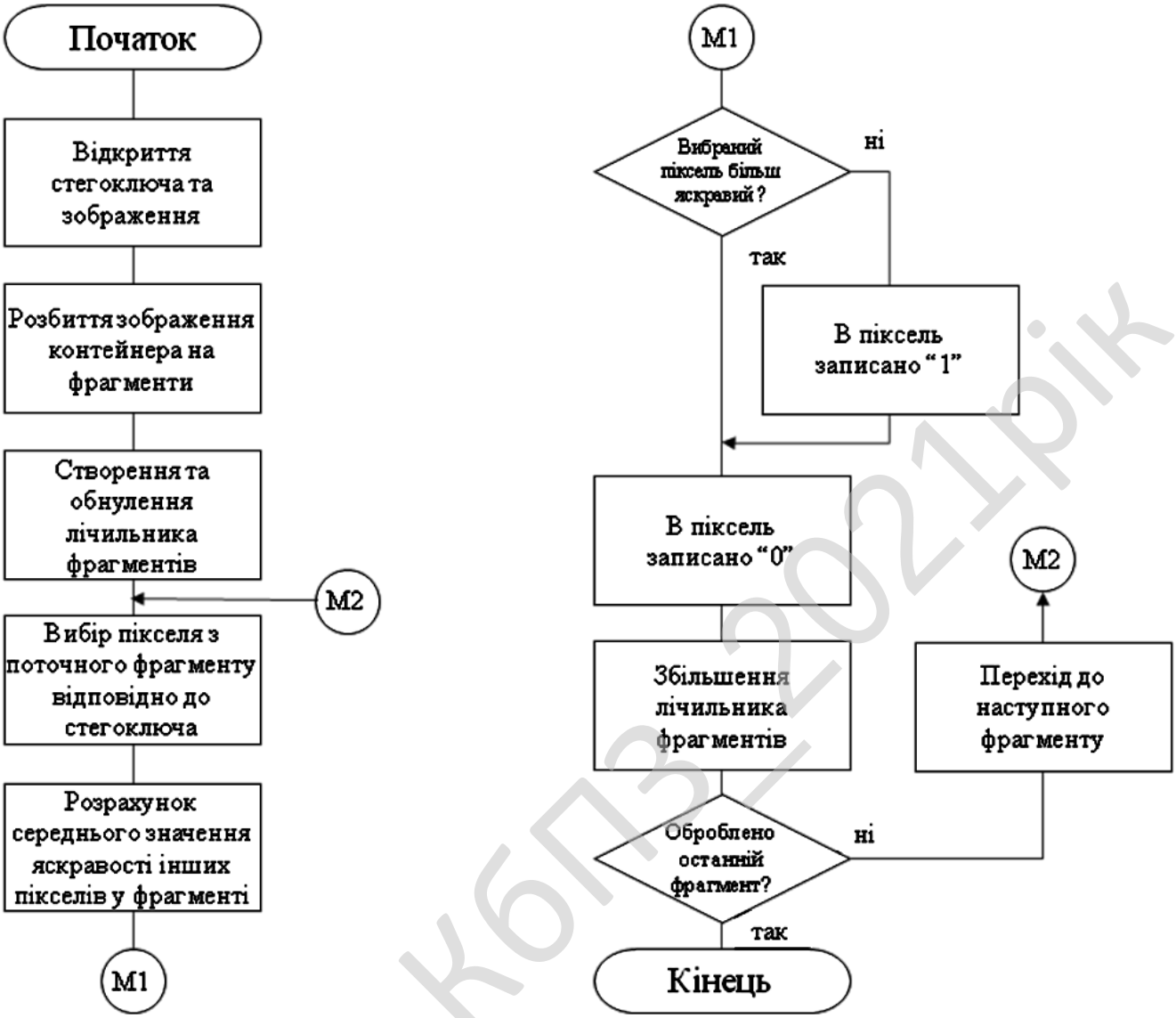

Рисунок 4.3 – Блок-схема роботи підпрограми читання стегоповідомлення

4.2 Захист розробленого програмного забезпечення

Розроблене програмне забезпечення захистимо за допомогою національного стандарту захисту інформації на основі алгоритму шифрування/дешифрування ДСТУ 4145-2002 з використанням еліптичних кривих над двійковим розширеним полем Галуа. У системі шифрування/дешифрування як параметри розглядається еліптична крива E_p (a,b) і точка G на ній. Учасник В вибирає закритий ключ n і обчислює відкритий ключ

$P_B = n \times G$. Щоб зашифрувати повідомлення P_m використовується відкритий ключ одержувача В P_B. Учасник А вибирає випадкове ціле позитивне число k і обчислює зашифроване повідомлення C_m, що є точкою на еліптичній кривій.

$$C_m = \{k \times G,\ P_m + k \times P_B\}. \quad (4.1)$$

Щоб дешифрувати повідомлення, учасник В множить першу координату точки на свій закритий ключ і віднімає результат від другої координати:

$$P_m + k \times P_B - n_B \times (k \times G) = P_m + k \times (n_B \times G) - n_B \times (k \times G) = P_m. \quad (4.2)$$

Учасник А зашифрував повідомлення P_m додаванням до нього $k \times P_B$. Ніхто не знає значення k, тому, хоча P_B і є відкритим ключем, ніхто не знає $k \times P_B$. Супротивнику для відновлення повідомлення доведеться обчислити k. Зробити це буде нелегко. Одержувач також не знає k, але йому як підказку посилається $k \times G$. Помноживши $k \times G$ на свій закритий ключ, одержувач одержить значення, що було додано відправником до незашифрованого повідомлення. Тим самим одержувач, не знаючи k, але маючи свій закритий ключ, може відновити незашифроване повідомлення.

Нехай задано просте число $p > 4$. Тоді еліптичною кривою E, визначеною над розширеним двійковим полем полем F_{2^m}, називається безліч пар чисел $(x, y), x, y \in F$, що задовольняють тотожності:

$$y^2 \equiv x^3 + a \cdot x + b (\mathrm{mod}\, 2^m), \quad (4.3)$$

де $4 \cdot a^3 + 27 \cdot b^2$ не рівно з нулю по модулю 2^m.

Інваріантом еліптичної кривої називається величина $J(E)$, що задовольняє тотожності:

$$J(E) \equiv 1728 \frac{4a^3}{4a^3 + 27b^2} (\mathrm{mod}\, 2^m). \quad (4.4)$$

Коефіцієнти a, b еліптичної кривої E, по відомому інваріанту $J(E)$, визначаються таким чином:

$$\begin{cases} a \equiv 3k (\mathrm{mod}\, 2^m) \\ b \equiv 2k (\mathrm{mod}\, 2^m) \end{cases} \text{де } k \equiv \frac{J(E)}{1728 - J(E)} (\mathrm{mod}\, 2^m), J(E) \neq 0 \text{ або } 1728. \quad (4.5)$$

Пари (x,y), що задовольняють тотожності (4.1), називаються точками

еліптичної кривої *E*, *x* та *y* – відповідно *x*– та *y*-координатами точки.

Точки еліптичної кривої позначатимемо *Q*(*x*,*y*) або просто *Q*. Дві точки еліптичної кривої рівні, якщо рівні їх відповідні *x*– і *y*-координати.

На безлічі всіх точок еліптичною к ривою Е введемо операцію додавання, яку позначатимемо знаком "+". Для двох довільних точок $Q_1(x_1, y_1)$ та $Q_2(x_2, y_2)$ еліптичної кривої Е, розглянемо декілька варіантів.

Нехай координати точок Q_1 та Q_2 задовольняють умові $x_1 \neq x_2$. В цьому випадку їх сумою називатимемо точку $Q_3(x_3, y_3)$ координати якої визначаються порівняннями:

$$\begin{cases} x_3 \equiv \lambda^2 - x_1 - x_2 (\mathrm{mod}\, 2^m), \\ y_3 \equiv \lambda(x_1 - x_3) - y_1 (\mathrm{mod}\, 2^m), \end{cases} \text{де } \lambda \equiv \frac{y_2 - y_1}{x_2 - x_1} (\mathrm{mod}\, 2^m). \qquad (4.6)$$

Якщо виконана рівність $x_1 = x_2$ та $y_1 = y_2 \neq 0$, то визначимо координати точки Q_3 таким чином:

$$\begin{cases} x_3 \equiv \lambda^2 - 2x_1 (\mathrm{mod}\, 2^m), \\ y_3 \equiv \lambda(x_1 - x_3) - y_1 (\mathrm{mod}\, 2^m), \end{cases} \text{де } \lambda \equiv \frac{3x_1^2 + a}{2y_1} (\mathrm{mod}\, 2^m). \qquad (4.7)$$

У разі, коли виконана умова $x_1 = x_2$ та $y_1 = -y_2 (\mathrm{mod}\, p)$, суму точок Q_1та Q_2 називатимемо нульовою точкою О, не визначаючи її х– і у-координати. В цьому випадку, точка Q_2 називається запереченням точки Q_1. Для нульової точки О виконана рівність:

$$Q + 0 = 0 + Q = Q, \qquad (4.8)$$

де *Q* – довільна точка еліптичної кривої Е.

Щодо введеної операції складання безліч всіх точок еліптичною кривою Е, разом з нульовою точкою, утворюють кінцеву абельову (комутативну) групу порядку т, для якого виконана нерівність:

$$p + 1 - 2\sqrt{p} \leq m \leq p + 1 + 2\sqrt{p} \qquad (4.9)$$

Точка *Q* називається точкою кратності *k*, або просто – кратною точкою еліптичної кривої *E*, якщо для деякої точки *P* виконана рівність:

$$Q = P + \ldots + P = kP \qquad (4.10)$$

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено головне вікно програми. Як можна побачити з рисунку головне вікно розподілено на підгрупи інформаційного виведення даних (зображення), верхнього меню програми, функціональних кнопок, поля введення стеганоповідомлень.

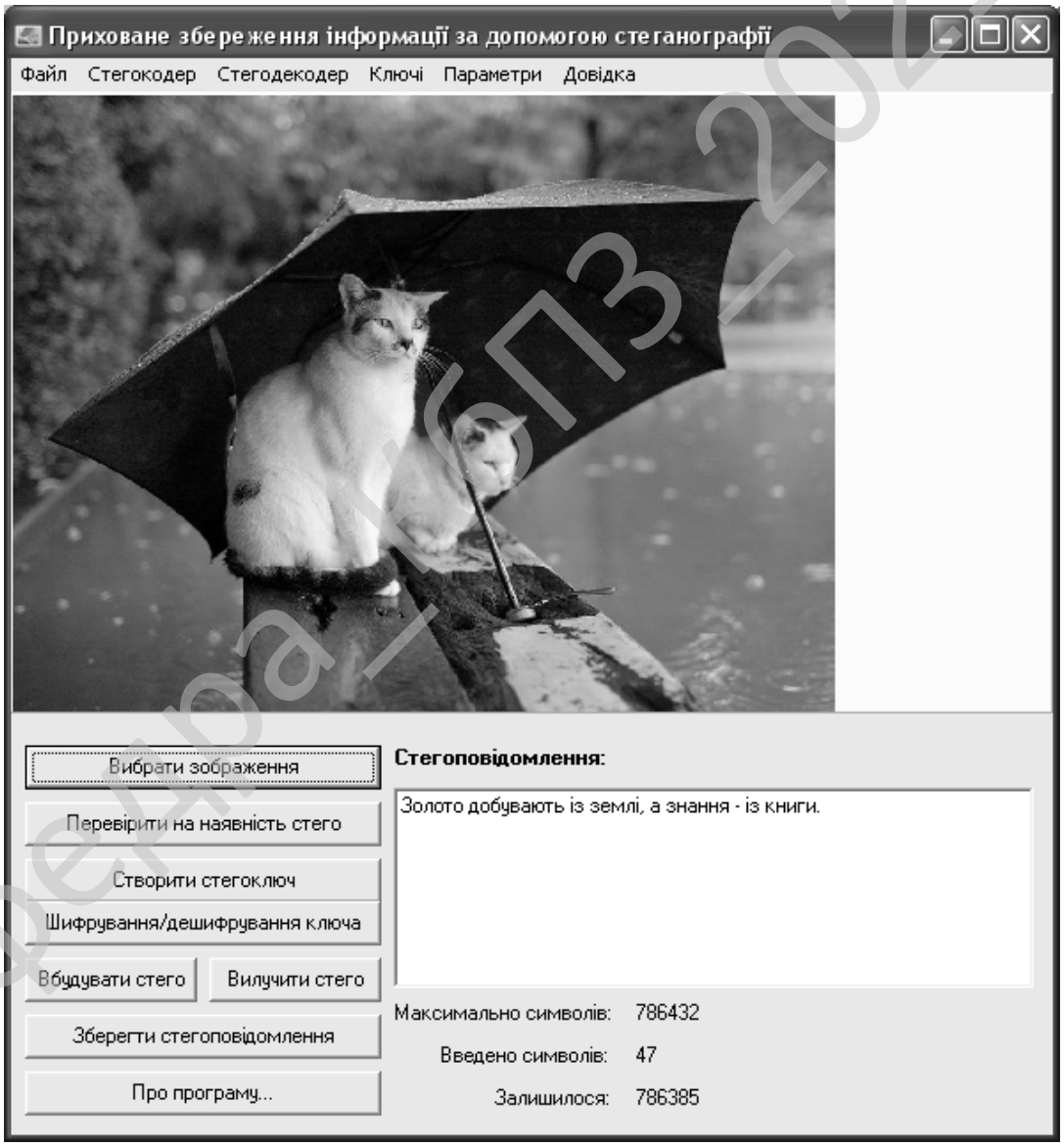

Рисунок 5.1 – Головне вікно програми

З нього видно, що програма дозволяє виконувати такі основні дії над зображенням:

– Проводити дії з меню файл.

– Проводити дії з меню Стегокодер.

– Проводити дії з меню Стегодекодер.

– Проводити дії з меню Ключі.

– Проводити дії з меню Параметри.

– Проводити дії з меню Довідка.

– Вибрати зображення.

– Перевірити на наявність стегоконтейнера.

– Створити стегоключ.

– Шифрувати стегоключ.

– Дешифрувати стегоключ.

– Вбудувати стегоконтейнер.

– Вилучити стегоконтейнер.

– Зберегти стегоповідомлення.

– Про програму.

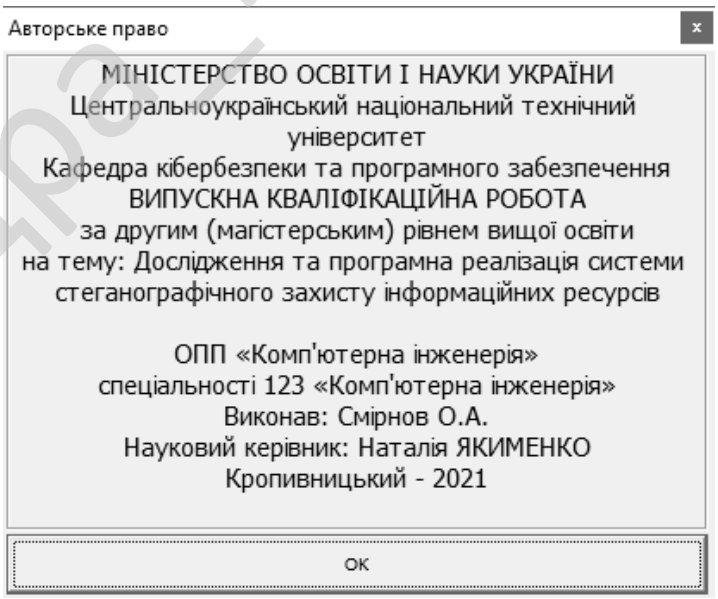

Рисунок 5.2 – Довідка

Крім того у програмі є вікно, куди виводяться повідомлення, які буди приховані та закодовані у зображені.

На рисунку 5.2 зображено вікно довідки, на якому наведено дані про розробника магістерської роботи та програмного продукту, керівника, й місце, де була виконана магістерська робота.

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи стеганографічного захисту інформаційних ресурсів.

Метою розробки є дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

Об'єктом дослідження є процес стеганографічного захисту інформаційних ресурсів.

Предметом дослідження є методи стеганографічного захисту інформаційних ресурсів.

Методи дослідження базуються на методах теорії захисту інформації, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод стеганографічного захисту інформаційних ресурсів.

– Розроблено вітчизняний продукт стеганографічного захисту інформаційних ресурсів, який має більш широкі можливості, на відміну від існуючих аналогів.

7 ДАНІ ПРО ЕКОНОМІЧНУ ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ

7.1 Техніко-економічне обґрунтування теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Після ознайомлення з підприємством та засобами розробки програмної продукції був розроблений план розробки програми. Був підрахований необхідний час для розробки та впровадження програми. Цей час склав 24 днів (один місяць). В магістерській роботі було проведено дослідження та виконана програмна реалізація системи стеганографічного захисту інформаційних ресурсів. Розроблене програмне забезпечення має достатню надійність і задовольняє усім поставленим умовам, а саме:

а) невеликий розмір;

б) невеликі системні потреби;

в) незалежність від встановлених на комп'ютері баз даних;

г) зручність у користуванні та надійність.

Таблиця 7.1 – Початкові дані

Показники	Позначення	Характеристика або величина
1	2	3
1. Кількість розроблених програм період, шт.	N	1
2. Кількість екземплярів програм, шт.	Ne	01 (2 ост. цифри № зал $*10^1$)
3. Запланований термін розробки, днів	Fpq	24 (1 місяць)
4. Група задачі підсистеми управління (1-6)	–	1
5. Ступінь новизни задачі (А, Б, В, Г)	–	Г
6. Складність алгоритму (1, 2, 3)	–	2

Продовження таблиці 7.1

1	2	3
7. Кількість макетів вхідної інформації	–	8
8. Кількість форм вихідної інформації.	–	6
9. Мова програмування (1-6)	–	2
10. Попередній досвід (1-6)	–	3
11. Гнучкість проекту ПП (1-6)	–	3
12. Детальність проекту ПП (1-6)	–	1
13. Рівень спрацьованості колективу (1-6)	–	2
14. Ступінь вимірності процесів (1-6)	–	3
15. Необхідна надійність програмного забезпечення (1-6)	–	3
16. Розмір бази даних (порівняно з розміром програми) (1-6)	–	4
17. Складність кінцевого програмного продукту (1-6)	–	5
18. Необхідний рівень забезпечення повторного використання (1-6)	–	2
19. Документованість відповідно до планованого життєвого циклу (1-6)	–	3
20. Вимоги до швидкодії ПП (1-6)	–	3
21. Обмеження на розміри основного сховища даних (1-6)	–	2
22. Різноманітність використовуваних обчислювальних платформ (1-6)	–	4
23. Професійний рівень аналітиків (1-6)	–	3
24. Професійний рівень програмістів (1-6)	–	4
25. Постійність складу команди розробників (1-6)	–	2
26. Досвід розробки додатків (1-6)	–	1
27. Досвід роботи з обчислювальною платформою (1-6)	–	2

Продовження таблиці 7.1

1	2	3
28. Досвід роботи з мовою і інструментами середовища розробки (1-6)	–	2
29. Досвід роботи з програмними інструментами розробки (1-6)	–	3
30. Розробка ПЗ для декількох серверів одночасно (1-6)	–	3
31. Вимоги до дотримання встановленого графіка робіт (1-6)	–	2
32. Вартість ПЗ у розробника (НМА), грн.	–	30000
33. Норматив додаткової зарплати, % :	$Н_д$	10
34. Норматив відрахувань у соціальні фонди, %	Нс	22
35. Норматив загальногосподарських витрат, %	Нг	15
36. Норматив витрат на освоєння нових мов програмування, %	Нп	15
37. Рівень рентабельності програмної продукції, %	Ре	40
38. Ставка податку на додану вартість, %	Ндв	20

7.2 Розрахунок трудомісткості розробки програмної продукції

Значення трудомісткості розробки програмного забезпечення для стадій ТЗ, ЕК, ТП та ВП визначаємо по типовим нормам часу приведеним в додатках МВ. Стадія РП є найбільш тривалою і трудомісткою, що робить значний вплив на інші стадії проекту.

Визначимо трудомісткість розробки ПЗ для стадії РП.

Обчислюємо номінальні трудовитрати, люд-міс.:

$$T_{ном} = A \, \text{Size}^B, \qquad (7.1)$$

де: A – коефіцієнт Боема, A = 2,45;

Size – загальний об'єм відлагодженого програмного коду, тис. рядків;

B – показник ступеня, що визначається співвідношенням:

$$B = 1{,}01 + 0{,}001\sum W_i, \qquad (7.2)$$

де: *Wi* – сумарне значення п'яти показників (МВ, додаток 2), що відображають особливості розробки проекту програмного продукту (ПП) і колективу розробників.

$$B = 1{,}01+0{,}001(2{,}43+3{,}64+4{,}22+3{,}95+2{,}73) = 1{,}027.$$

$$T_{ном} = 2{,}45 \cdot 2{,}2^{1{,}027} = 5{,}5 \text{ люд-міс.}$$

Визначаємо уточнені (з урахуванням приведених в МВ додатку 3 сімнадцяти додаткових коефіцієнтів) трудовитрати, люд-міс.:

$$T_{уточн} = T_{ном}\, ПВ_j, \qquad (7.3)$$

де: $ПВ_j$ – добуток сімнадцяти додаткових коефіцієнтів, приведених в МВ додатку 3.

$$T_{уточн} = 5{,}5 \cdot (1 \cdot 1{,}09 \cdot 1{,}30 \cdot 0{,}91 \cdot 1 \cdot 1 \cdot 1 \cdot 1{,}15 \cdot 1 \cdot 0{,}87 \cdot 1{,}10 \cdot$$

$$\cdot 1{,}22 \cdot 1{,}12 \cdot 1{,}10 \cdot 1 \cdot 1 \cdot 1{,}10) = 12{,}9 \text{ люд-міс.}$$

Ці коефіцієнти дозволяють диференційовано оцінювати результати роботи програмістів, беручи до уваги швидкодію програми, використання різноманітних обчислювальних платформ і інструментів розробки, взаємодію декількох серверів, вимоги до об'ємів баз даних і ін.

Визначаємо підсумкові трудовитрати по стадії робочий проект, люд-дні:

$$T_{РП} = 0{,}3 C T_{уточн}^{0{,}33+0{,}2(B-1{,}01)} S, \qquad (7.4)$$

де: *C* – визначений емпірично коефіцієнт, запропонований авторами методики, (МВ, додаток 4);

S – коефіцієнт стиснення (або подовження) графіка робіт %, що дозволяє коректувати терміни розробки ПЗ згідно встановленим вимогам. Вибираємо в межах (25…350)%.

$$T_{РП} = 0{,}3 \cdot 2{,}66 \cdot 12{,}9^{0{,}33+0{,}2(1{,}027-1{,}01)} \cdot 100 = 131 \text{ люд/день.}$$

Для зручності визначення загальної трудомісткості на розробку програмного забезпечення результати розрахунків по стадіям зводимо до таблиці 7.2.

Таблиця 7.2 – Визначення трудомісткості розробки програмного забезпечення

Стадії розробки	Трудомісткість за типовими нормами та розрахунками	
	Величина, люд/дні	Підстава
Технічне завдання	9	Д5
Ескізний проект	10	Д6
Технічний проект	15	Д7
Робочий проект	131	Ф 7.1-7.4
Впровадження	15	Д13
Всього	180	–

7.3 Визначення чисельності виконавців і планового фонду зарплати

Чисельність ставок інженерів-програмістів для розробки програмного забезпечення визначається за формулою:

$$Ч = \frac{T_{пз} N}{F_{рq} - H_{ев}}, \tag{7.5}$$

де: $F_{рq}$ – плановий фонд робочого часу одного спеціаліста, днів;

$T_{пз}$ – трудомісткість розробки програмного забезпечення люд-дні.

$$Ч = \frac{180 \cdot 1}{24 - 3} = 8{,}6 \text{ ставки.}$$

Чисельність інженерів-електронщиків для проведення технічного обслуговування та ремонту комп'ютерних мереж визначається в залежності від наявності технічних засобів і норм витрат часу на виконання профілактичних робіт на протязі року.

Визначаємо затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за період розробки. Результати розрахунку зводимо до таблиці 7.3.

Таблиця 7.3 – Затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за розрахунковий період

Найменування обладнання	Профілактичне обслуговування			
	Кількість хв. на один. обл.	Кількість облад-нання	Затрати часу в хв.	Затрати часу в год.
Системний блок ПК	385	12	4620	77
Монітор	160	12	1920	32
Клавіатура	140	12	1680	28
Маніпулятор «мишка»	30	12	360	6
Принтер матричний	185	1	185	3
Принтер лазерний	355	2	710	12
Принтер струминний	300	1	300	5
Сканер	155	2	310	5
Концентратор– маршрутизатор	155	2	310	5
Кабельні господарства ЛОМ на 1 м. п.	2,5	70	175	3
Кабельне господарство електромережі	48	50	2400	40
Копіювальний апарат	285	1	285	5
Усього за рік:			$З_ч$	221

Час на профілактику обладнання в загальному балансі робочого часу інженерів-електронщиків неповинен складати більше 10%.

Виходячи з цього фонд робочого часу інженерів-електронщиків складає:

$$Ф_{др}^{с} = \frac{З_ч \cdot n_{міс}}{1,2}, \quad (7.6)$$

$$Ф_{др}^{с} = \frac{221 \cdot 1}{1,2} = 184,1 \text{ год.}$$

Визначаємо необхідну кількість ставок штатного персоналу сектора ТО:

$$Ч_{ел} = \frac{Ф^{с}_{др}}{F_{др} \cdot T_{зм}}, \quad (7.7)$$

$Ч_{ел}$ = 184,1/(24·8) = 1 ставки.

Для забезпечення нормального технічного обслуговування засобів ТО та мереж, необхідно прийняти найбільше ціле значення розрахункової чисельності інженерів-електронщиків.

Чисельність інженерів-системотехніків, адміністраторів мережі, дизайнерів WEB вузлів, системних програмістів (аналітиків), бухгалтерів-економістів визначається за потребою в залежності від функціональних обов'язків. Після визначення чисельності персоналу складається штатний розклад.

Таблиця 7.4 – Розрахунок чисельності штатного персоналу сектору системного та адміністративного обслуговування засобів ОТ та комп'ютерних мереж

Посада	Вид роботи	Час	К-ть штатних одиниць
Адміністратор загальної мережі, аналітик	Адміністрування локальної мережі, поштового та серверу DNS (ОС FreeBSD), маршрутизатора Cisco, доменного контролеру Windows Server 2008 R2, серверу доступу ADSL (ОС Linux), налаштування ADSL, VPN, PPPoE, Frame Relay, Wi-Fi	1	0,5
	Налаштування і конфігурування базової станції безпровідного зв'язку (CMTS)	1	
	Розробка та впровадження проектів з організації зв'язку між віддаленими об'єктами, ЛОМ	1	
	Забезпечення цілодобової роботи зв'язку клієнтів до мережі Інтернет	1	
Всього		4	

Продовження таблиці 7.4

Посада	Вид роботи	Час	К-ть штатних одиниць
Продакт-менеджер	Презентації нової продукції, пошук каналів збуту	1	0,5
	Підтримка постійних клієнтів	1	
	Оформлення договорів, ведення тендерів	1	
	Контроль взаєморозрахунків з постачальниками	1	
Всього		4	
Дизайнер WEB	Розробка концепції оформлення та інтерфейсу сайту, оптимізація дизайну існуючих, проектує їх структуру та навігацію	1	0,5
	Створення графічних і стилістичних елементів сайту	1	
	Оформлення банерів і промо-сторінок	1	
	Розміщення графіки і контенту на Інтернет сторінках	1	
Всього		4	
Інженер верстальник	Розробка та верстка макетів рекламної продукції та технічної документації	1	0,5
	Верстка друкованих видань	1	
	Додрукова підготовка макетів	1	
	Розміщення графіки і контенту на Інтернет сторінках	1	
Всього		4	

Складемо штатний розклад виконавців.

Таблиця 7.5 – Штатний розклад виконавців

Посада	Кількість ставок	Середньомісячний оклад, грн.	Всього за період розробки, грн.
Керівник (ІТ-менеджер)	1	10500	10500
Продакт-менеджер	0,5	8500	4250
Інженер-програміст	8,6	7500	64500
Інженер-електронщик	1	6000	6000
Інженер-системотехнік	0,5	6000	3000
Адміністратор мережі	0,5	8000	4000
Дизайнер WEB	0,5	10000	5000
Всього за період розробки	$R_{сп}$ = 12,6	-	$Ф_{роб}$ = 97250

Розрахуємо середньоденну зарплату одного виконавця:

$$З_{сд} = \frac{Ф_{роб}}{R_{сп} F_{рд}}, \quad (7.8)$$

де: $Ф_{роб}$ – загальна сума зарплати за плановий період, грн.

$$З_{сд} = \frac{97250}{12,6 \cdot 24} = 322 \; грн.$$

7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника

Балансова вартість будівель визначається з урахуванням кількості робочих місць виконавців, питомої площі на одне робоче місце, та вартості одного квадратного метра виробничої площі:

$$Б_{уд} = R^1_{сп} S_у Ц_{пл}, \quad (7.9)$$

де: $R^1_{сп}$ – кількість робочих місць виконавців, шт. Приймаємо 13 робочих місць;

$S_у$ – питома площа на одне робоче місце, $м^2$;

$Ц_{пл}$ – вартість одного квадратного метра площі, грн.

Згідно даних ТОВ науково-дослідницького консалтингового підприємства «Пекторaль» (м. Кіровоград) ціна одного квадратного метра площі новобудови, вік якої не перевищує 25 років, по місту складає 800…1600 $у.о./м^2$. Враховуючи, що курс складає 1 у.о. = 25 грн. приймаємо для розрахунку вартість одного метра квадратного рівною 20000 грн./м^2. На кожне робоче місце у середньому потрібно 8 $м^2$. З урахуванням цього:

$$Б_{уд} = 13 \cdot 8 \cdot 20000 = 2080000 \text{ грн.}$$

Вартість передавальних пристроїв складає 10% від вартості будівель, і у даному випадку вона складе: 208000 грн. Балансова вартість інвентарю розраховується за нормою 3500 грн. на одне робоче місце. Тобто:

$$I_{нв} = R^1_{сп} \cdot Ц_м, \qquad (7.10)$$

де: $Ц_м$ – ціна меблів для одного робочого місця, грн.

$$I_{нв} = 13 \cdot 3500 = 45500 \text{ грн.}$$

Балансова вартість обчислювальної техніки визначається по оптовим цінам постачальника з врахуванням витрат на транспортування.

Специфікація на обчислювальну техніку наведена в таблиці 7.7.

Дані по оптовій ціні на обладнання та комплектуючі вибирались по прайсу фірми Brain за 24.10.20 – джерело http: //brain.com.ua.

Таблиця 7.6 – Специфікація

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Персональний комп'ютер		11457
Системний блок		7509
Процесор	Intel Core i7-4790 (4(8) ядра по 3.6 – 4.0 GHz); Cache Memory 8 MB	4200

Продовження таблиці 7.6

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Системна плата	1st Player ATX NEW	1525
Відеокарта	PCIeX: ATI HD5670 SAPPHIRE 1024MB/128bit/DDR3/TV/DualDVI	430
Жорсткий диск	HDD: 500 Gb 7200 Serial ATA WD 16MB	490
Оперативна пам'ять	Kingston DDR3 2GB (KVR1333D3N9/2G) Intel/AMD – 2 шт	333
DVD-привод	-	-
Корпус	ATX Middle Tower GIGABYTE GZ-X4 Silver 500W (GZ-X4 Silver)	411
Кулер	–	–
Кардрідер внутрішній	USB 2.0 Card reader STORM CR-35U1A4-B, int. 3.5", 1*USB2.0+AUDIO+1394, multi: All Type Cards, black	120
інше	Клавіатура, мишка	Подару-нок
Монітор	22" TFT, ASUS VW223D (5ms, 300/3000: 1, 170/160, D-SUB, Wide)	2600
Принтер лазерний	Canon i-SENSYS LBP6030W	2700
Принтер струминний	Epson Stylus Photo P50 (C11CA45341) + USB cable	5500
Сканер	Epson Perfection V37 Photo	2970
Копіювальний апарат	Canon i-SENSYS MF217W with Wi-Fi	5965
Пристрій безперебійного живлення	UPS APC BACK-UPS ES 525VA 230V RUSSIA (BE525-RS)	1348

Витрати на транспорт, монтаж та випробування можуть бути прийняті в межах до 10% від оптової ціни.

Для визначення необхідної кількості капітальних вкладень складемо таблицю 7.8.

Таблиця 7.7 – Балансова вартість обчислювальної техніки

Найменування обчислювальної техніки	Кількість, шт.	Ціна за одиницю, грн.	Витрати на транспортування, монтаж та випробовування.	Загальна вартість, грн.
Персональні комп'ютери	13	11457	14894,1	163835,1
Принтер лаз.	2	2700	540	5940
Принтер струм.	1	5500	550	6050
Сканери	1	2970	297	3267
Копіюв. апарат	1	5965	596,5	6561,5
Всього	–	–	–	185653,6

Таблиця 7.8 – Вартість основних фондів та амортизаційні відрахування розробника

Групи та види основних фондів	Балансова вартість, грн.	Амортизація	
		Норма, %	Відрахування, грн.
1	2	3	4
Група 3			
1. Будівлі	2080000	-	-
2. Передавальні пристрої	208000	-	-
Всього по групі	2288000	5	114400

Продовження таблиці 7.8

1	2	3	4
Група 4			
3. Обчислювальна техніка	185654	-	-
Всього по групі	185654	50	92827
Група 5, 6			
4. Вимірювальні пристрої	3999	25	-
5. Транспортні засоби	0	20	-
6. Господарський інвентар	45500	25	-
Всього по групі	49499	-	12374,75
7. Нематеріальні активи	30000	10	3000
Разом	$К_р = 2553153$		$А_р = 222602$

7.5 Визначення собівартості розробки та ціни програмної продукції

Визначимо основну зарплату виконавців:

$$З_о = \frac{З_{сд} \cdot Т_{пз}}{N_e}, \quad (7.11)$$

де: N_e – кількість екземплярів програм, шт.

$$З_о = 322 \cdot 180/30 = 1932 \text{ грн.}$$

Визначимо додаткову зарплату (оплата відпусток, виконання державних та суспільних обов'язків) на рівні 10%:

$$З_д = З_о \cdot Н_q \cdot 0{,}01, \quad (7.12)$$

де: $Н_q$ – норматив додаткової зарплати, %.

$$З_д = 1932 \cdot 10 \cdot 0{,}01 = 193 \text{ грн.}$$

Відрахування на соціальні потреби за нормативом $Н_с = 22\%$ від суми основної та додаткової зарплати:

$$C_{оц} = 0{,}01 \cdot H_c (З_о + З_д), \quad (7.13)$$

де: H_c – відрахування на соціальні потреби, %.

$$C_{оц} = 0{,}01 \cdot 22(1932+193) = 468 \text{ грн.}$$

Визначимо загальногосподарські витрати (електроенергію, ремонт і утримання приміщень і т.д) за нормативом $H_г$ = 15% від основної зарплати:

$$Г_{осп} = З_о \cdot H_г \cdot 0{,}01, \quad (7.14)$$

де: $H_г$ – загальногосподарські витрати, %.

$$Г_{осп} = 1932 \cdot 15 \cdot 0{,}01 = 290 \text{ грн.}$$

Визначимо витрати на матеріали для розробки програмної продукції за нормами споживання та діючими цінами за одиницю виміру:

$$З_М = (З_{М1} + З_{М2} + З_{М3})/N_е, \quad (7.15)$$

де: $З_{М1}$ – вартість паперу, грн.;

$З_{М2}$ – вартість запам'ятовуючих пристроїв, грн.;

$З_{М3}$ – вартість фарби, картріджей, тонеру, грн.;

$N_е$ – кількість екземплярів програм, шт.

Згідно виданих викладачем норм n = 0,33 приймаємо одну пачку паперу на три місяці розробки. Тоді, враховуючи, що вартість пачки паперу складає $Ц_п$ = 121 грн., визначаємо вартість паперу за період розробки $N_м$ = 1 міс:

$$З_{М1} = Ц_п \cdot N_м \cdot n. \quad (7.16)$$

$$З_{М1} = 121 \cdot 1 \cdot 0{,}33 = 40 \text{ грн.}$$

Згідно виданих викладачем норм до вартості запам'ятовуючих пристроїв входить вартість CD дисків в кількості, що дорівнює кількості екземплярів програм та одного DVD диска для збереження резервної копії програми:

$$З_{М2} = \sum Ц_д, \quad (7.17)$$

де: $Ц_д$ – вартість дисків CD/DVD: CDR TDK 700Mb, 80Min, 52x Cake box – 3 грн./шт., DVD-R LG 4,7Gb, 16x speed Cake box – 3 грн./шт.

$$З_{М2} = 30 \cdot 3 + 3 = 93 \text{ грн.}$$

Згідно виданих викладачем норм одноразовій заправці підлягають усі друкуючі пристрої і становить:

$$З_{МЗ} = \sum Ц_{з}., \quad (7.18)$$

де: $Ц_з$ – вартість розхідних матеріалів друкуючих пристроїв: відновлення та заправка картриджу для Canon i-SENSYS LBP6030W – 574 грн.; картридж для Epson Stylus Photo P50 – 558 грн.; відновлення картриджу для MF217W – 570 грн.

$$З_{МЗ} = 574+558+570 = 1702 \text{ грн.}$$

$$З_{М} = (40 + 93 + 1702)/30 = 61 \text{ грн.}$$

Визначимо витрати на освоєння нових мов програмування або операційних систем за нормативом ($Н_п$ = 15%) від основної зарплати виконавців:

$$О_п = З_о \cdot Н_п \cdot 0{,}01, \quad (7.19)$$

де: $Н_п$ – норматив витрат на освоєння нових мов програмування, %.

$$О_п = 1932 \cdot 15 \cdot 0{,}01 = 290 \text{ грн.}$$

Визначимо витрати на амортизацію основних фондів з урахуванням загальної річної суми амортизаційних відрахувань та кількості екземплярів програм ($N_е$ = 30 прим.):

$$А_м = \frac{А_р \cdot N_{міс}}{N_е \cdot 12}, \quad (7.20)$$

де: $А_р$ – загальна річна сума амортизаційних відрахувань, грн.

$$А_м = 222602 \cdot 1/(30 \cdot 12) = 618 \text{ грн.}$$

Повна собівартість ПЗ визначається як сума витрат за попередніми статтями калькуляції:

$$С_п = З_о + З_д + С_{оц} + Г_{осп} + З_м + О_п + А_м. \quad (7.21)$$

$$С_п = 1932+193+468+290+61+290+618 = 3852 \text{ грн.}$$

Величини ціна підприємства, податок на додану вартість, відпускна ціна програмної продукції визначаються за формулами, приведеними в таблиці 7.9

Таблиця 7.9 – Нормативна калькуляція собівартості розробки програмного забезпечення задачі

Найменування статей витрат	Позначення	Величина, грн.
1	2	3
1. Основна зарплата виконавців	$З_о$	1932
2. Додаткова зарплата виконавців	$З_д$	193
3. Відрахування на соціальні потреби	$С_{оц}$	468
4. Загальногосподарські витрати	$Г_{осп}$	290
5. Витрати на матеріали	$З_М$	61
6. Освоєння нових операційних систем, мов програмування	$О_п$	290
7. Амортизація основних фондів	$А_м$	618
8. Повна собівартість програмного забезпечення	$С_п$	3852
9. Плановий прибуток	$П_р$	1541
10.Ціна підприємства $Ц_п = С_п + П_р$	$Ц_п$	5393
11.Податок на додану вартість $ПДВ = 0.01 \cdot Н_{дв} \cdot Ц_п$	$ПДВ$	1077
12.Відпускна ціна програмної продукції $Ц = Ц_п + ПДВ$	$Ц$	6470

Визначимо плановий прибуток за рівнем рентабельності ($Р_п$) програмної продукції, яка залежить від складності програми та ступеня новизни задачі.

Для даного програмного забезпечення рівень рентабельності складає 40%.

$$П_р = 0{,}01 \cdot Р_п \cdot С_п, \qquad (7.22)$$

де: $Р_п$ – рівень рентабельності, %.

$$П_р = 0{,}01 \cdot 40 \cdot 3852 = 1541 \text{ грн.}$$

7.6 Визначення об’єму капітальних вкладень у споживача програмної продукції

Об’єм капітальних вкладень у споживача програмної продукції визначаємо на основі балансової вартості основних фондів, яка враховує ціну, транспортно-заготівельні витрати, вартість будівель, монтажних та пусконалагоджувальних робіт, а також витрати на випробування у виробничих умовах. Результати розрахунків зводимо у таблицю 7.10.

Таблиця 7.10 – Розрахунок об’єму капітальних вкладень у споживача програмної продукції

Найменування капітальних вкладень	Сума за варіантами, грн.	
	Базовий	Новий
Вартість програмної продукції	–	6470
Всього капітальних витрат	–	6470

7.7 Визначення експлуатаційних витрат

Експлуатаційні витрати у споживача програмної продукції визначаємо при умові роботи підсистеми на протязі року. Результати зводимо до таблиці 7.11.

Витрати на профілактичні роботи:

$$З_р = Т_р \cdot З_г \cdot (1 + 0{,}01 \cdot Н_q) \cdot (1 + 0{,}01 \cdot Н_c), \qquad (7.23)$$

де: $Т_р$ – кількість годин обслуговування кожного комп’ютера за рік, *год.*;

$З_г$ – заробітна плата обслуговуючого персоналу, *грн / год*.

Таблиця 7.11 – Розрахунок експлуатаційних витрат у споживача програмної продукції

Найменування статей витрат	Позначення	Сума витрат за варіантами, грн.	
		Базовий	Новий
1. Витрати на технічне обслуговування	$З_р$	32208	20130
2. Витрати на електроенергію	$З_{ел}$	22680	22050
3. Витрати на амортизацію	$З_{ам}$	0	3235
Всього витрат за рік	$І$	54888	45415

Після купівлі нового програмного забезпечення кількість профілактичних годин робіт зменшилася з 240 годин на рік до 150 годин на рік, тому витрати на технічне обслуговування зменшилися з:

$$З_{р\,баз} = 240 \cdot 100 \cdot 1{,}1 \cdot 1{,}22 = 32208\ грн,$$

до:

$$З_{р\,нов} = 150 \cdot 100 \cdot 1{,}1 \cdot 1{,}22 = 20130\ грн.$$

Витрати на електроенергію визначаються з урахуванням споживаємої потужності ($П_{ел}$) в кіловатах, часу експлуатації технічних засобів ($Т_р$) в годинах та ціни однієї кіловат-години ($Ц_{ел}$):

$$З_{ел} = П_{ел} \cdot Т_р \cdot Ц_{ел}. \qquad (7.24)$$

$$З_{ел\,баз} = 10 \cdot 0{,}15 \cdot 7200 \cdot 2{,}1 = 22680 \text{ грн.}$$

$$З_{ел\,нов} = 10 \cdot 0{,}15 \cdot 7000 \cdot 2{,}1 = 22050 \text{ грн.}$$

Витрати по амортизації визначаються на основі норм амортизаційних відрахувань, вартості програмної продукції і основних фондів. Для розрахунку складаємо таблицю 7.12.

Таблиця 7.12 – Розрахунок амортизаційних відрахувань

Групи основних фондів	Норма амортизації %	Балансова вартість, грн., за варіантами		Сума відрахувань, грн., за варіантами	
		Базовий	Новий	Базовий	Новий
Програмна продукція	50	–	6470	–	3235
Всього відрахувань	-	–	6470	–	3235

7.8 Визначення економічної ефективності програмної продукції

Економічна ефективність програмного забезпечення визначається для виготовлювача і споживача за такими показниками.

Величина економічного ефекту при виготовленні програмної продукції, розраховуємо за формулою:

$$E_в = (Ц_п - C_п) \cdot N_е - \sum_{i=1}^{m} E_{p_m} \cdot K_{p_m}, \quad (7.25)$$

де: $K_р$ – балансова вартість основних фондів розробника, грн.; $E_р$ – розрахунковий коефіцієнт капіталовкладень.

$$Ев = (5393 - 3852) \cdot 30 - (0,05 \cdot 2288000 + 0,5 \cdot 185654 + +0,25 \cdot 49499 + 0,1 \cdot 30000) \cdot 1/12 = 27680 \ грн.$$

Визначимо період окупності додаткових капітальних вкладень у виробника програмної продукції:

$$T_в = \frac{K_p^*}{(Ц_п - C_п) \cdot N_е}, \quad (7.26)$$

де: K_p^* – балансова вартість основних фондів розробника без врахування вартості ОФ третьої групи, так як їх строк служби на порядок більший ніж період розробки ПЗ.

$$T_в = \frac{265153}{(5393 - 3852) \cdot 30 \cdot 12 / 1} = 0,5 \ року.$$

Таблиця 7.13 – Показники економічної ефективності програмної продукції

Найменування показників	Одиниця виміру	Величина
1. Кількість екземплярів програми	Прим.	30
2. Повна собівартість розробленої програми	Грн.	3852
3. Ціна розробленої програми	Грн.	5393
4. Плановий прибуток від реалізації розробленої програми	Грн.	1541
5. Рентабельність програмної продукції	%	40
6. Об'єм додаткових капітальних вкладень у виробника програмної продукції	Грн.	2553153
7. Загальний прибуток від реалізації програмної продукції	Грн.	46230
8. Величина економічного ефекту при виготовлені програмної продукції	Грн.	27680
9. Період окупності додаткових капітальних вкладень у виробника програмної продукції	Років	0,5
10.Об'єм додаткових капітальних вкладень у споживача програмної продукції	Грн.	6470
11.Величина економічного ефекту у користувача програмної продукції	Грн.	6238
12.Період окупності додаткових капітальних вкладень у користувача програмної продукції	Років	0,7

Визначимо величину економічного ефекту у користувача програмної продукції за формулою:

$$E_{сп} = (I_б - I_н) - E_н(K_н - K_б), \qquad (7.27)$$

де: $I_б$, $I_н$ – величина експлуатаційних витрат за базовим и новим варіантом відповідно;

$K_б$, $K_н$ – об'єм капітальних вкладень за варіантами, що порівнюються.

$$E_{сп} = (54888\text{-}45415)\text{-}0,\ 5 \cdot 6470 = 6238 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у споживача програмної продукції за рахунок зниження експлуатаційних витрат:

$$T_{сп} = \frac{K_н - K_б}{I_б - I_н}, \tag{7.28}$$

$$T_{сп} = \frac{6470}{54888 - 45415} = 0,7 \ року .$$

Показники економічної ефективності програмної продукції зводимо до таблиці 7.13.

7.9 Висновки

Розроблена програма економічно вигідна. За рахунок впровадження програмного забезпечення досягається скорочення часу обробки інформації, підвищується культура праці, підвищення якості приймаючих управлінських рішень.

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Шкідливі і небезпечні фактори при роботі з комп'ютером

Протягом усієї історії людство приділяє прискіпливу увагу безпеці життя. Охорона праці є складовою частиною безпеки життя.

Законом України "Про охорону праці" регламентуються загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби польців та м'язи рук) при роботі з клавіатурою мають теж істотне навантаженням. До шкідливих факторів, які впливають на робітників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

При розгляді шкідливих чинників роботи програмістів та інших спеціалістів ІТ будемо керуватись наступними нормативно-правовими актами: «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98, та «Правила охорони

праці під час експлуатації електронно-обчислювальних машин» НПАОП 0.00-1.28-10,

Умови праці програміста вулючають наступні фактори:

– параметри повітряного середовища в приміщенні;

– вентиляція приміщення;

– освітлення приміщення;

– параметри повітряного середовища в приміщенні, тощо.

Щоб запропонувати заходи щодо зменшення впливу комп'ютера на організм програміста визначемо фактори, які можуть викликати професійне захворювання і впливають на працездатність програміста,

Програміст працює з електронно-обчислювальною машиною (ЕОМ) та іншим обладнанням, яке є джерелом небезпеки ураження електричним струмом. Так як робота програміста характеризується істотним зоровим навантаженням, то вимагає належного освітлення. Так як програміст постійно перебуває в приміщенні, тому для комфортних умов праці в цьому приміщенні необхідно створити належний мікроклімат.

При роботі з використанням ЕОМ відзначають наступні небезпечні та шкідливі фактори:

– ризик виникнення надзвичайних ситуацій природного або штучного характеру на об'єкті або території.

– ризик виникнення пожежі;

– негативний вплив на органи зору людини;

– ризики ураження електричним струмом;

– недостатня, або надмірна освітленість робочого місця;

– монотонність праці;

– електромагнітні (у т.ч. високочастотні) електромагнітні випромінювання (коливання);

– несприятливі мікрокліматичні умови;

– нервово-емоційна напруженість праці;

– інтелектуальні навантаження;

– невідповідність ергономічних показників робочого місця діючим вимогам;

– шуми;

– статичні навантаження на кістково-м’язовий апарат;

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	8,5*
Довжина	13*
Висота	2,9

* вказано загальні розміри поєднаного приміщення, де загалом працюють 16 людей, а фактично у наявності є дві кімнати, розділених перестінком.

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,9
Обсяг, V	$м^3$	не менше 20.0	20

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

У зазначеному приміщенні працює 16 осіб. За даними, які наведено у табл. 8.1 та табл. 8.2, можна зробити висновок, що площа та об’єм приміщення у розрахунку на одно робоче місце програміста відповідають нормативним вимогам

(Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»).

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, яка виконується в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря у приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-23	40-55	0,1
Тепла	23-25	50-70	0,1	23-24	55-70	0,14

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер *HP Laser 135a*, електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

Працю працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5 – 28 – 2006 р. можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V -го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному

висвітленні повинна становити 300 лк. Крім того все поле зору повинне бути освітлено достатньо рівномірно – ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.3 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень;

Таким чином можна припустити, що основною причиною можнивого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов’язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

8.4 Розрахункова частина

Початкові данні для розрахунку захисного заземлення.

Тип заземлення: робоче заземлення нульової точки трансформатора. Заземленню підлягає виробниче обладнання фірми. Напруга – 220/380 В.

Розрахунок проводиться для випадку розміщення заземлюючого пристрою в однорідній землі за допустимим опором розтіканню струму заземлювача методом коефіцієнта використання заземлювачів.

Для заземлення застосовуються: ветикальні електроди – пруток діаметром D=12 мм, довжиною L = 2,5 м. Відстань між вертикальними заземлювачами A=2,5 м. Розміри перетину з'єднуючої полоси: 40 х 4 мм. Глибина закладення контура заземлення t= 0,7 м. Товщина верхнього шару грунта H=1,7 м. Ширина (діаметр) горизонтального заземлювача: b=40 мм. Відстань від центра вертикального заземлювача до поверхні замлі: T=1,95 м. Сезонний кліматичний коефіцієнт вертикального заземлювача: Cv=1,4. Сезонний кліматичний коефіцієнт горизонтального заземлювача: Cv=1,5. Питомий опір верхнього шару грунта (суглинок): ρ_1= 100 Ом*м. Питомий опір нижнього шару грунта (супесь): ρ_2= 150 Ом*м. Тип вертикального заземлювача: пруток діаметром 12 мм. Тип горизонтального заземлювача (з'єднуючої полоси): полоса. Розташування заземлювачив: по контуру. Опір заземлювача, який нормується: R_{3H} = 4 Ом. Коефіцієнт використання вертикального заземлювача η_C = 0,55 (табличне значення, залежить від кількості вертикальних стержнів та співвідношення A/L). Коефіцієнт використання горизонтального заземлювача=0,34 (табличне значення, залежить від співвідношення A/L).

Розрахунок захисного заземлення можна автоматизувати за допомогою програми (автор О.В.Оришака), сирцевий код якої опублікован на стр.13-16 Метододичних вказівок до виконання розрахунків з викор. персон. ЕОМ IBM сумісного типу / Кіровоград. ін-т с.-г. машинобуд. ; [укл. О. В. Оришака, Є. К. Солових, В. О. Оришака]. – Кіровоград : КІСМ, 1997. – 20 с., URL :

http://dspace.kntu.kr.ua/jspui/handle/123456789/4358. Остання була використана мною для автоматизації процесу розрахунку.

Розрахунок.

Еквівалентний питомий опір:

$$\rho_{екв} = \psi \rho_1 \rho_2 L / [\rho_1 \psi(L-H+t) + \rho_2 \psi(H-t)] = 112,9 \text{ Ом.}$$

Опір одного вертикального заземлювача:

$$R_O = [\rho_1/(2\pi*L)]*[(ln)2L/D)+0,5ln[(4T+L)/(4T-L)] = 45,77 \text{ Ом.}$$

Опір, який нормується, якщо $\rho_{екв} > 100$ (у нас $\rho_{екв} = 112,9 > 100$):

$$R = R_{ЗН} * \rho_{екв}/100 = 4*112,9/100 = 4,529 \text{ Ом.}$$

Опір розтіканню горизонтального заземлювача (полоси):

$$R_П = 0,366(\rho_{екв} \psi / L_П \cdot \eta_П) \lg(2 / L_П * L_П / bt) = 30,03 \text{ Ом.}$$

Де довжина горизонтального заземлювача (полоси):

L=A*n =18*2,5=45 м. (при розташуванні заземлювачів по контуру).

Де n – ітераційна кількість вертикальних заземлювачів.

Опір розтіканню вертикальних заземлювачів:

$$R_B = R_П R / (R_П - R) = 4,61 \text{ Ом.}$$

Кількість вертикальних заземлювачів:

$$n = R_O / R_B * \eta_C = 18 \text{ шт.}$$

8.5 Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва вцілому.

З цих міркувань було здійснено аналіз приміщення, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров'я програміста. Розроблено заходи з охорони праці.

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи стеганографічного захисту інформаційних ресурсів.

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

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів стеганографічного захисту інформаційних ресурсів.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем стеганографічного захисту інформаційних ресурсів.

– Досліджена система стеганографічного захисту інформаційних ресурсів.

– На основі отриманих результатів досліджень створена програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання стеганографічного захисту інформаційних ресурсів.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

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. 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 й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows XP/Vista/7/8/10.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм ДСТУ 4145-2002.

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.

Розроблена програма має реальний економічний ефект від її впровадження у виробництво у сумі 6238 грн. З урахуванням вартості розробки програми та обладнання, строк окуплення становить 0,7 роки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Смірнов О.А. Дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів // Збірник праць молодих науковців ЦНТУ. – Вип. 12. – Кропивницький: ЦНТУ, 2022.

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

3. *Смирнов А.А.* Разработка метода и алгоритмов синтеза больших ансамблей недвоичных дискретных сигналов на основе обобщенных перестановочных преобразований / А.А. Кузнецов, Ал.М. Носик, А.А. Смирнов, Л.Н. Качур, Ан.М. Носик // Збірник наукових праць «Системи обробки інформації». – Випуск 5(72). – Х.: ХУПС – 2008. – С. 151-156.

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

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

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

2012. – С. 273-281.

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

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

9. *Смирнов А.А.* Встраивание данных в контейнеры-изображения с использованием сложных дискретных сигналов / А.А. Кузнецов, А.А. Смирнов // Радиотехника: Всеукраинский межведомственный научно-технический сборник. Тематический выпуск «Информационная безопасность» – Выпуск 166. – Х.: ХНУРЭ. – 2011. – С. 134-141.

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

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

12. *Smirnov O.A.* Use of Complex Discrete Signals for Steganographic Information Security / A.A. Kuznetsov, A.A. Smirnov // International Journal of

Engineering Practical Education. – Volume 1, Issue 1. – USA, Indiana, Riley: Science and Engineering Publishing Company. – 2012. – P. 21-25.

13. Заявка на патент МПК H04L 9/00. Спосіб стеганографічного приховування та вилучення даних в просторовій області зображень із використанням прямого розширення спектра та пристрій для його реалізації (варіанти) Смірнов О.А., Смірнов О.А.; заявник та патентоволодар Харківський національний університет радіоелектроніки. – №a201208110; заявл. 02.07.2012. (повідомлення про завершення формальної експертизи за заявкою на винахід № 23092/ЗА/12 від 04.10.2012).

14. Звіт з держбюджетної теми № 36Б113 «Розробка методів підвищення оперативності передачі та захисту інформації у телекомунікаційних системах» (№ держреєстрації 0113U003086). Керівник Смірнов О.А. К.: КНТУ. – 2013. – 88 с.

15. Звіт з науково-дослідницької роботи «Формування псевдовипадкових послідовностей для підвищення оперативності передачі та захищеності інформації у телекомунікаційних системах» (№ держреєстрації 0112U002598). Керівник Смірнов О.А. К.: КНТУ. – 2012. – 85 с.

16. Звіт з науково-дослідницької роботи «Розробка стеганографічних засобів вбудовування інформації в нерухливі та рухливі зображення» (№ держреєстрації 0112U002599). Керівник Смірнов О.А. К.: КНТУ. – 2012. – 67 с.

17. Звіт з науково-дослідницької роботи «Розробка методів підвищення безпеки телекомунікаційних мереж» (№ держреєстрації 0112U006630). Керівник Смірнов О.А. К.: КНТУ. – 2012. – 74 с.

18. Звіт з науково-дослідницької роботи «Методи підвищення оперативності передачі даних та захисту інформації у телекомунікаційній мережі» (№ держреєстрації 0112U006631). Керівник Смірнов О.А. К.: КНТУ. – 2012. – 95 с.

19. Звіт з держбюджетної (кафедральної) науково-дослідної теми № 101/14.01.2006 «Методи та моделі стеганографічного захисту інформації від кібератак». Керівник Корченко О.Г. К.: НАУ. – 2006. – 132 с.

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

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

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

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

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

25. *Смірнов О.А.* Аналіз та дослідження стеганографічних систем та протоколів для захисту інформації та інформаційних ресурсів / О.А. Смірнов // Збірник тез III міжнародної науково-практичної

конференції «Інформаційні технології та комп'ютерна інженерія». м. Вінниця. 29-31 травня 2012 р. – Вінниця: ВНТУ. – 2012. – С. 164-166.

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

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

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

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

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

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

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

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

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

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

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

37. *Смірнов О.А*. Стеганографічне приховування даних в аудіо-контейнерах із використанням прямого розширення спектру / О.А. Смірнов // Збірник тез XIX науково-практичної конференції «Проблеми створення, розвитку та застосування інформаційних систем

спеціального призначення». м. Житомир. 19 квітня 2012 р. – Житомир: ЖВІ НАУ. – 2012. – С. 147-148.

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

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

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

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

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

43. *Смирнов А.А.* Исследование известных методов синтеза дискретных сигналов / А.А. Смирнов // Наука і техніка Повітряних Сил Збройних Сил України. – Випуск 3(9). – Х.: ХУПС. – 2012. – С. 123-126.

44. *Смирнов А.А.* Исследование методов синтеза дискретных сигналов с

особыми корреляционными свойствами / А.А. Смирнов // Збірник наукових праць «Системи обробки інформації». – Випуск 9(107). – Х.: ХУПС – 2012. – С. 81-85.

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

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

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

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

49. *Смирнов А.А.* Синтез больших ансамблей дискретных сигналов с использованием недвоичных избыточных кодов / А.А. Смирнов, А.М. Носик, Л.Н. Качур, С.Ю. Стасев // Матеріали IV наукової конференції Харківського Університету Повітряних Сил імені Івана Кожедуба. Харків. 16-17 квітня 2008 р.– Харків. ХУПС. – 2008. – С. 154.

50. *Смирнов А.А.* Дискретні сигнали з багаторівневою функцією кореляції / А.А. Смирнов // Збірник тез XII міжнародного науково-практичного

семінару «Комбінаторні конфігурації та їх застосування». м. Кіровоград. 14-15 жовтня 2011 р. – Кіровоград: КНТУ. – 2011. – С. 119-120.

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

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

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

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

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

56. *Смирнов А.А.* Обоснование предложений по реализации динамического режима функционирования радиосистем управления с множественным доступом / А.А. Смирнов, В.Н. Сай, А.В. Коваленко // Системи управління,

навігації та зв'язку. – Випуск 3(23). – К.: ДП «ЦНДІНУ». – 2012. – С. 255-262.

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

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

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

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

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

62. *Стасєв Ю.В.* Основи теорії побудови сигналів / Ю.В. Стасєв. – Х.: ХВУ, 1999. – 87с.

63. *Стасев Ю.В.* Использование сложных дискретных сигналов для стеганографической защиты информации / Ю.В. Стасев, А.А. Кузнецов, А.А. Смирнов // Системи управління, навігації та зв'язку. – Випуск 3 (19). – К.: ДП «ЦНДІНУ». – 2011. – С. 110-114.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

					ВКРМ-123.21.0001.00.00.ТЗ		
Вим.	Арк.	№ документа	Підпис	Дата			
Розробив		Смірнов О.А.			Дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів	Літ.	Аркуш 1 / Аркушів 6
Перевірив		Якименко Н.М.				М	
Н. Контр.		Гермак В.С.				ЦНТУ КІ-20М3	
Затв.		Смірнов О.А.					

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи стеганографічного захисту інформаційних ресурсів.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 41-13 від 02.08.2021 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи стеганографічного захисту інформаційних ресурсів.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) 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 з ОС та з користувачем;

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи стеганографічного захисту інформаційних ресурсів;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

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

5.4 Вимоги до архітектури

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

5.5 Вимоги до надійності

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

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

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

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows XP/Vista/7/8/10 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 XP/Vista/7/8/10.

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 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2021 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинні бути розглянуті Шкідливі і небезпечні фактори при роботі з комп’ютером.

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 3 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 99 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

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

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 10.12.2021 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 22.12.2021 р.

Додаток Б

(обов'язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

другим (магістерським) рівнем вищої освіти

_____ Якименко Н.М.

Дослідження та програмна реалізація

системи стеганографічного захисту інформаційних ресурсів

Лістинг програми

Код документу 12

Носій: CD/DVD-диск

Загальна кількість аркушів: 28

Літера: РП

Кропивницький – 2021 року

MAINSTEGANOGRAPHIC.PAS – основна программа

```
unit mainsteganographic;

interface

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

type

  TSteganographicgraphy = 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;
    MainsteganographicMenu1: TMainsteganographicMenu;
    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
  Steganographicgraphy: TSteganographicgraphy;

implementation

uses password_steganographic, Hamming;

{$R *.DFM}

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

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

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

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

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

end.
```

STEGANOGRAPHIC.PAS – реалізація алгоритму стеганографії

```
unit Steganographic;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, password_steganographic, 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 ща межу
         begin
            inc (curline);
            pt:=form2.bmp.ScanLine[curline];  // вказівник на наступний рядок
            curcol:=1;
         end;
         // вбудовування інформації
         if ((pt^[curcol] and 1) <> 0) then dat:=dat or (1 shl (x-1)) else
dat:=dat and (not((1 shl (x-1))));
         inc (curcol);
      end;
      dat_^[y]:=byte(dat);
   end;
   decode(dat_,len,totpoz);
   totpoz:=totpoz+len;
end;

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

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

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

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

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

```
end;

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

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

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

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

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

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

procedure TForm2.N1Click(Sender: TObject);
var
   f : file;
   s,s1 : string;
begin
   if not ldfl.execute then exit;
   assignfile(f,ldfl.filename);
   filemode:=0;
   if ioresult <> 0 then ;
   {$I-}
   reset(f,1);
   if ioresult <> 0 then
   begin
      application.messagebox('Неможливо відкрити вказаний файл','',$10);
      exit;
   end;
   inc (ft2cnt);
   if estlen < filesize(f) then
   begin
      application.messagebox('Не 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.
```

PASSWORD_STEGANOGRAPHIC.PAS - модуль створення стегоключа

```
unit password_steganographic;

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.
```

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('ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ
ВИЩОЇ ОСВІТИ');
Memo1.Lines.Add('');
Memo1.Lines.Add('на  тему:');
Memo1.Lines.Add('');
Memo1.Lines.Add('Дослідження та програмна реалізація системи стеганографічного
захисту інформаційних ресурсів');
Memo1.Lines.Add('');
Memo1.Lines.Add('Керівник:  Якименко Н.М.');
Memo1.Lines.Add('');
Memo1.Lines.Add('Розробив:  студент   Смірнов Олексій Анатолійович');
Memo1.Lines.Add('           гр. КІ-20МЗ');
Memo1.Lines.Add('');
Memo1.Lines.Add('м. Кропивницький 2021');
Memo1.Lines.Add('');
end;

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