Міністерство освіти і науки України
Одеський національний політехнічний університет
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення

Бохонько Михайло Володимирович,
студент групи РЗ-151

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Модифікація методу хеш-стеганографії, заснованого на передачі послідовності цифрових зображень

Напрям:
125
Кібербезпека

Керівник:
Зоріло Вікторія Вікторівна,
к.т.н., ст. викл.

Одеса – 2020

Міністерство освіти і науки України
Одеський національний політехнічний університет
Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій
Кафедра кібербезпеки та програмного забезпечення

Рівень вищої освіти: другий (магістерський)
Спеціальність: 125 - Кібербезпека
Освітня програма - КІбербезпека

ЗАТВЕРДЖУЮ
Завідувач. кафедри ІУЗІС

_____ Кобозєва А.А.

«____» _____ 2020р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Бохонька Михайла Володимировича, РЗ-151

1. Тема проекту (роботи) Модифікація методу хеш-стеганографії, заснованого на передачі послідовності цифрових зображень.
   Керівник роботи: к.т.н., ст. викл. Зоріло В.В.

   затверджені наказом ректора від „16” листопада 2020р. № 468-в
2. Зміст пояснювальної записки: аналіз відомих засобів для обчислення перцептивних хеш-кодів зображень, реалізація існуючих хеш-алгоритмів та адаптація їх для задачі хеш-стеганографії, модифікація алгоритму хеш-стеганографії на основі проведеного аналізу, розробка програмного інтерфейсу для алгоритмічної реалізації запропонованого методу.

3. Перелік графічного матеріалу: схема одноразової хеш-стеганографії, схема відповідності «символ-зображення», схема робочого приміщення, схема розташування в грунті заземлення.

4. Консультанти розділів роботи

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Охорона праці та безпека в надзвичайних ситуаціях	Ярова І.А. к.т.н., доцент	05.11.2020	26.11.2020

5. Дата видачі завдання “_____”_____20___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання	Примітка
1	*Аналіз літератури з теми квалфікаційної роботи*	01-09-2020	*виконано*
2	*Модифікація методу хеш-стеганографії*	01-10-2020	*виконано*
3	*Розробка програмного інтерфейсу*	16-11-2020	*виконано*
4	*Підготовка тексту роботи*	02-11-2020	*виконано*
5	*Підготовка презентації та доповіді*	18-12-2020	*виконано*
6	*Попередній захист*	01-12-2020	*виконано*
7	*Нормоконтроль, рецензування*	15-12-2020	*виконано*
8	*Занесення роботи в електронний архів*	25-12-2020	*виконано*
9	*Допуск до захисту у завідувача кафедри*	25-12-2020	*виконано*

Здобувач вищої освіти _____ Бохонько М.В.

Керівник роботи _____ Зоріло В.В.

ЗАВДАННЯ

на розробку розділу “Охорона праці та безпека в надзвичайних ситуаціях”

Бохонько Михайлу Володимировичу, група РЗ-151

Інститут інформаційної безпеки, радіоелектроніки та телекомунікацій

Кафедра кібербезпеки та програмного забезпечення

Тема роботи *Модифікація методу хеш-стеганографії, заснованого на передачі послідовності цифрових зображень*

Зміст розділу:

1 Аналіз умов праці і вибір основних заходів захисту від небезпечних і шкідливих виробничих факторів.

2 Аналіз техногенних небезпек, вибір заходів і засобів забезпечення безпеки у надзвичайних ситуаціях.

3 Розрахунок штучного заземлення.

Керівник роботи	Консультант з охорони праці
_____ (_____)	_____ (_____)
«____» _____ 2020 р.	«____» _____ 2020 р.

АНОТАЦІЯ

Кваліфікаційна робота на тему «Модифікація методу хеш-стеганографії, заснованого на передачі послідовності цифрових зображень» на здобуття освітньо-кваліфікаційного рівня “Магістр” з спеціальності 125 – «Кібербезпека» містить 20 рисунків, 5 таблиць, 2 додатків, 50 літературних джерел за переліком посилань. Робота виконана на 82 сторінках загального тексту і 52 сторінках основного тексту.

Метою даної роботи є підвищення ефективності передачі секретного повідомлення шляхом модифікації метода хеш-стеганографії.

У результаті виконання кваліфікаційної роботи розроблено модифікацію методу хеш-стеганографії та створено програмний продукт для кодування та декодування повідомлення із стеганоповідомлення. Проведено аналіз результатів нового методу та порівняння їх із початковим, після чого зроблено висновок, що модифікований метод краще забезпечує стійкість стеганоповідомлення до різних атак.

Результати даної роботи можуть бути використані для захисту текстової інформації при передачі по каналу зв’язку від несанкціонованого її використання сторонніми особами.

СТЕГАНОГРАФІЯ, ХЕШ-КОД, ЦИФРОВЕ ЗОБРАЖЕННЯ, ХЕШ-СТЕГАНОГРАФІЯ.

ANNOTATION

Qualification work on "Modification of hash-steganography method based on digital image sequence transfer" for educational qualification level "Master" from specialty 125 - "Cybersecurity" contains 20 figures, 5 tables, 2 appendices, 50 references sources according to the list of references. Work is executed on 82 pages of the general text and 52 pages of the basic text.

The aim of this work is to increase the efficiency of the transmission of secret messages by modifying the method of hash steganography.

As a result of the qualification work a modification of the method of hash steganography was developed and a software product for encoding and decoding messages with steganopovidolenie was created. The results of the new method were analyzed and compared with the original method, after which it was concluded that the modified method better provides steganopovidolenie resistance to various attacks.

The results of this work can be used to protect textual information during transmission over a communication channel from unauthorized use by unauthorized persons.

STEGANOGRAPHY, HASH, DIGITAL IMAGE, HASH STEGANOGRAPHY.

ЗМІСТ

ВСТУП

Сьогодні стеганографія являє собою сукупність методів і технічних рішень, що реалізують захист інформації, заснований на різних принципах. Однак в умовах стрімкого зростання інформаційно-телекомунікаційних технологій найбільш активно розвиваються комп'ютерні методи стеганографії й способи їхнього застосування в кібернетичному просторі.

В основі багатьох підходів до вирішення завдань стеганографії лежить спільна із криптографією методична й інструментальна база, закладена Шеноном при розробці загальної теорії секретного зв'язку. Це пов'язано з тим, що стеганографія й криптографія розвивалися в рамках єдиної науки - тайнопису. Лише наприкінці XIX століття, після формулювання Кирхгофом базових законів криптографії, основний з яких полягав у тому, що стійкість криптографічного перетворення визначається таємністю ключа, криптографія відокремилася від стеганографії й стала розвиватися як самостійна наука. Визначальним моментом у стеганографії залишилося збереження в таємниці алгоритму стеганографічного перетворення.

Розвиток комп'ютерної стеганографії відбувається завдяки інтенсивному впровадженню в усі сфери діяльності людини засобів обчислювальної техніки й створенню широких можливостей для оперативного обміну різною інформацією у вигляді текстів, програм, звуку, відео й образів між будь-якими учасниками мережевих сеансів незалежно від їхнього територіального розміщення. Це дозволяє активно використовувати всі переваги, які дають стеганографічні методи.

Безпечна передача даних в Інтернеті була мрією з моменту появи Інтернету. Хеш-стеганографія є одним з засобів передачі даних, приховуючи інформацію в даних. Дані, які використовуються для приховування інформації у хеш-стеганографії, це зображення. Кілька алгоритмів для хеш-стеганографії, таких як, Chaos, міжнародний алгоритм шифрування даних (IDEA), розширений стандарт шифрування (AES), стандарт шифрування даних (DES), повідомлення Digest 5 (MD5), шифри потоку бітів (BSC), безпечний алгоритм хешу (SHA), пропонуються

для використання. Деякі з цих алгоритмів, такі як AES, DES і IDEA, є корисними для невеликої кількості даних, але не дуже корисні для масивних наборів даних, оскільки ці алгоритми включають інтенсивні обчислення та необхідні машини швидкої обробки. Аналогічно, інші алгоритми, такі як MD5, SHA, засновані на криптографічних хеш-функціях, які використовують хеш-ключ 16 байт. Але ці алгоритми є вразливими з точки зору забезпечення безпеки через властиві недоліки, спричинені підходом контрольної суми.

Розроблений новий алгоритм хеш-стеганографії, що описаний у роботі, є досить актуальним, адже його без зайвих проблем можна використовувати для приховування таємної інформації в медіа-контенті та передавати по каналам зв'язку. Порівняно з іншими методами в даній області, цей алгоритм є більш простішим та стійким по відношенню до інших алгоритмів.

У перспективі цього сценарію пропонуємо новий підхід, який використовує надійність досконалого алгоритму хеш-функції, яка є більш безпечною та ефективною.

Метою роботи є підвищення надійності передачі стеганографічного повідомлення шляхом модифікації алгоритму хеш-стеганографії.

Для досягнення поставленої мети потрібно вирішити наступні задачі:

- провести огляд відомих алгоритмів хеш-стеганографії;
- обґрунтувати вибір алгоритму для модифікації;
- реалізувати програмно модифікований алгоритм;
- провести аналіз його ефективності.

Об'єктом дослідження є перцептивні хеш-алгоритми.

Предметом дослідження є метод хеш-стеганографії.

Для досягнення поставленої у кваліфікаційній роботі мети, були використані чисельні методи, методи обробки цифрових зображень.

Розробка складається з чотирьох розділів. У першому розділі наведений короткий опис предметної області та аналіз різних перцептивних алгоритмів. У другому розділі детально розглянуто метод та запропоновано його модифікація алгоритмами, наведених у першому розділі. Третій розділ – реалізація програмного

продукту, у якому описано програмне середовище програмування, безпосередньо розробку програмного коду та інтерфейс користувача. Четвертий розділ – «Охорона праці», у якому наведений аналіз умов праці і вибір основних заходів виробничої безпеки на робочому місці користувача персонального комп'ютеру та аналіз пожежної безпеки і вибір заходів і засобів пожежної безпеки на робочому місці користувача персонального комп'ютеру.

Практичне значення кваліфікаційної роботи полягає в доведенні запропонованої модифікації хеш-стеганографічного методу до його алгоритмічної реалізації та отриманні високих показників стійкості до атак, таких як стиск, масштабування, розтягнення, накладання шуму, зміна яскравості та контрасту зображення.

Публікації. Матеріали кваліфікаційної роботи магістра опубліковані в [1].

1 ПЕРЦЕПТИВНІ ХЕШ-АЛГОРИТМИ

Інформація, яка передається по каналам зв’язку, піддається загрозам викриття, спотворенню та знищенню. Одним з можливих рішень проблеми загрози розкриття є використання стеганографічних методів.

Існуючі стеганографічні алгоритми лише частково відповідають вимогам, які пред’являються до систем прихованої передачі[2-3]. Стеганографія - це міждисциплінарна наука і мистецтво передавати приховані дані, всередині інших даних, які не приховані.

Актуальною є задача пошуку нових алгоритмів стеганографічного вбудовування інформації. В процесі вивчення даної проблеми, була виявлена тенденція методів вбудови повідомлення у графічні зображення.

Одним з таких методів являється алгоритм хеш-стеганографії. Ідея хеш-стеганографії полягає в тому, що в стеганоконтейнер нічого не вбудовуємо. Алгоритм методу досить легкий.

1. Беремо велику кількість зображень та визначаємо їх хеш-коди. Хеш-код або хеш – це результат перетворення вхідного масиву даних довільної довжини у вихідний бітовий ряд фіксованої довжини (результат застосування хеш-функції).
2. Наступним кроком хешуємо вхідне повідомлення, результат ділимо на рівні частини.
3. Кожну частину хеш-коду повідомлення порівнюємо з хеш-кодом зображень, якщо таких зображень декілька, то беремо любе. Отриману послідовність зображень передаємо в канал зв’язку.

Проблема методу полягає в тому, шо він не стійкий до атак, в результаті чого хеш-код зміниться і повідомлення буду втрачене. Для вирішення цієї проблеми було вирішено інтегрувати перцептивний хеш-алгоритм для визначення хеш-коду зображення.

Існує декілька перцептивних хеш-алгоритмів для роботи з зображеннями. Зазвичай побудова хешу складається з трьох основних стадій:

1. Попередня обробка. На цій стадії зображення приводиться до вигляду, в якому його легше обробляти для побудови хешу. Це може бути застосування різних фільтрів (наприклад, фільтр Гауса), знебарвлення, зменшення розмірів зображення і т.д.

2. Основні обчислення. З отриманого на першій стадії зображення будується матриця (або вектор). Матриця (вектор) може представляти собою матрицю частот (наприклад, після перетворення Фур'є), гістограму яскравостей, або ще більш спрощене зображення.

3. Побудова хешу. З матриці (вектора), отриманої на другій стадії беруться деякі (можливо всі) коефіцієнти і перетворюються в хеш. Зазвичай хеш виходить розміром від 8 до ~ 100 байт. Отримані показники хешів порівнюють за допомогою функцій, що обчислюють «відстань» між двома хешами.

Нижче розглянемо докладніше чотири перцептивних хеш-алгоритму: Average Hash[4], Discrete Cosine Transform Based Hash [4, 5], Radial Variance Based Hash [6, 7] і MarrHildreth Operator Based Hash [4, 8].

1.1 Хеш на основі середнього значення

Ідея алгоритму полягає у відображенні середнього значення низьких частот. У зображеннях високі частоти забезпечують деталізацію, а низькі - показують структуру. Тому для побудови такої хеш-функції, яка для схожих збережень видаватиме близький хеш, потрібно позбутися від високих частот. Етапи алгоритму наведені нижче.

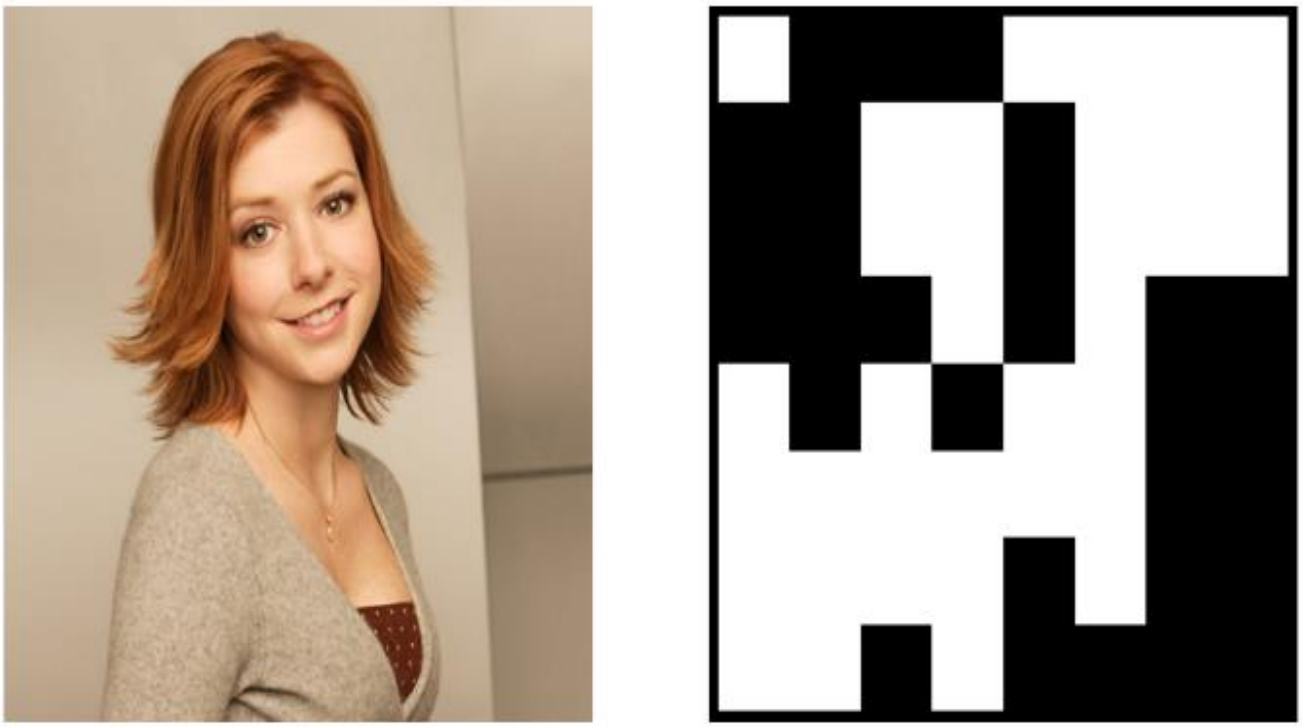
1. Зменшити розмір. Змінюємо розміри нашого зображення до розміру 8×8. Найшвидший спосіб позбутися від високих частот - зменшити зображення. Завдяки цьому ще більше спрощуємо наступні етапи, не втрачаючи занадто багато структурної інформації зображення, а також отримуємо деяку міру інваріантності масштабу.

2. Прибрати колір. Перетворити наше вхідне зображення у відтінки сірого. Цей крок значно підвищує продуктивність за рахунок скорочення обсягу

інформації, яку потрібно обробляти на більш пізніх етапах. Маленьке зображення переводиться в градації сірого, так що хеш зменшується втричі: з 64 пікселів (64 значення червоного, 64 зеленого ті 64 синього) всього до 64 значень кольору.

3. Знайти середнє. Обчислити середнє значення кольору для всіх 64 пікселів. Робимо це шляхом підсумовування всіх значень нашого зображення і ділення результату на розмір зображення

4. Побудувати ланцюжок бітів. Для кожного пікселя зображення робиться заміна кольору. Якщо значення пікселю більше середнього, то замість значення кольору ставимо 1. Аналогічно, якщо значення пікселю менше середнього – ставимо 0 (рис.1.1).



a) Початкове зображення b) Отриманий «відбиток»

Рисунок 1.1 – Приклад використання алгоритма Simple Hash

5. Побудувати хеш. Записуємо значення отриманого бітового зображення у певній послідовності, в результаті чого отримаємо значення довжиною 128 біт. Послідовність зчитування не має значення, але зазвичай біти записуються зліва направо, зверху вниз.

Отриманий хеш розміром 8 байт стійкий до масштабування, стиску або розтягування зображення, зміни яскравості, контрасту, маніпуляціями з кольорами. Головне достоїнство алгоритму – швидкість роботи.

1.2 Хеш-функція на основі дискретно-косинусного перетворення

Ідея даного методу полягає у розрахунку середнього значення за допомогою дискретно-косинусного перетворення (далі ДКП) для усунення високих частот.

ДКП – одне з ортогональних перетворень, тісно пов'язане з дискретним перетворенням Фур'є (далі ДПФ), і є гомоморфизмом його векторного простору. ДКП, як і будь-яке орієнтоване перетворення Фур'є, висловлює функцію або сигнал (послідовність з кінцевого числа точок даних) у вигляді суми синусоїд з різними частотами і амплітудами[9].

ДКП використовує тільки косинусні функції, на відміну від ДПФ, яке використовує і косинусні, і синусні функції. Існує вісім типів ДКП [10]. Найпоширеніший - другий тип. Його і будемо використовувати для побудови хеш-функції.

Визначення 1.1 (Другий тип ДКП):

Визначимо другий тип ДКП, як :

$$X[n] = \sum_{m=0}^{N-1} c[n,m] \cdot x[m], (n, m = 0, \dots, N-1), \quad (1.2)$$

де $c[n,m]$ – елемент матриці ДКП на перетині рядка з номером n та стовпця з номером m;

N – довжина сигналу;

$x[m]$ – послідовність сигналу довжиною N.

Визначення 1.2 (ДКП-матриця).

ДКП-матриця визначається як:

$$c[n,m] = \sqrt{\frac{2}{N}} \cdot \sum_{m=0}^{N-1} cos\left(\frac{(2m+1)\cdot n\pi}{2N}\right), (m,n = 0,\dots,N-1). \quad (1.3)$$

Рівняння 1.2 дуже зручне, коли ДКП обчислюється програмним шляхом. ДКП-матриця (1.3) може бути обчислена заздалегідь для будь-якої необхідної довжини. Таким чином, ДКП може бути представлено у вигляді:

$$\text{ДКП(I)} = \text{M} \cdot \text{I} \cdot \text{M}', \quad (1.4)$$

де *M* – ДКП-матриця;

I – зображення квадратного розміру, яке співпадає з розміром M;

M′ - зворотна матриця.

Різні властивості ДКП можуть бути використані для створення перцептивних хеш-функцій. Низькочастотні коефіцієнти ДКП найбільш стабільні до маніпуляцій з зображеннями. Це відбувається тому, що більша частина інформації сигналу, як правило, зосереджена в декількох низькочастотних коефіцієнтах. Ця властивість використовується, наприклад, при стисненні зображення за допомогою стандарту JPEG.

В якості матриці *I* зазвичай береться зображення, спрощене (за допомогою різних фільтрів, наприклад, знебарвлення) і стислий до розміру 32×32. В результаті виходить матриця ДКП (*I*), в лівому верхньому кутку якої і знаходяться низькочастотні коефіцієнти.

Щоб побудувати хеш потрібно взяти лівий верхній блок частот 8×8. Потім з цього блоку будується хеш розміром 8 байт за допомогою знаходження середнього і побудови ланцюжка біт.

Отже, наведемо кроки побудови хешу за допомогою даного алгоритму.

1. Прибрати колір. Для усунення непотрібних високих частот.
2. Застосувати медіанний фільтр [11]. Для зменшення рівня шуму. Суть медіанного фільтра в тому, що зображення розбивається на так звані «вікна», потім кожне вікно замінюється медіаною [12] для сусідніх вікон.

3. Зменшити зображення до розміру 32×32.
4. Застосувати ДКП до зображення.
5. Побудувати хеш.

Головні переваги такого хешу: стійкість до розмиття і стиснення зображення, а також швидкість порівняння хешів, завдяки їхньому маленькому розміру.

1.3 Хеш-функція на основі радіальної дисперсії

Ідея алгоритму Radial Variance Based Hash полягає в побудові променевого вектору дисперсії (далі ПВД) на основі перетворення Радону. Потім до ПВД застосовується ДКП і обчислюється хеш.

Перетворення Радону - це інтегральне перетворення функції багатьох змінних вздовж прямої[13-14].

Воно стійке до обробки зображень за допомогою різних маніпуляцій (наприклад, стиснення) і геометричних перетворень (наприклад, поворотів). У двовимірному випадку перетворення Радону для функції f(x, y) виглядає так:

$$R(s,a) = \int_{-\infty}^{\infty} f(s\cos\alpha - z\sin\alpha\,, s\sin\alpha + z\cos\alpha\,)dz. \qquad (1.5)$$

Перетворення Радона має простий геометричний сенс – це інтеграл від функції вздовж прямої, перпендикулярної вектору $\vec{n} = (\cos\alpha\,, \sin\alpha)$ і проходить на відстані s (виміряного уздовж вектора $\vec{n}$, з відповідним знаком) від початку координат.

Щоб перетворення Радона розширити для дискретних зображень, лінійний інтеграл по прямій $d = x \cdot \cos\alpha + y \cdot \sin\alpha$ може бути апроксимірован шляхом підсумовування значень всіх пікселів, що лежать в лінії шириною один піксель (рис. 1.2).

$$d - \frac{1}{2} \leq x \cdot \cos\alpha + y \sin\alpha \leq d + \frac{1}{2}$$

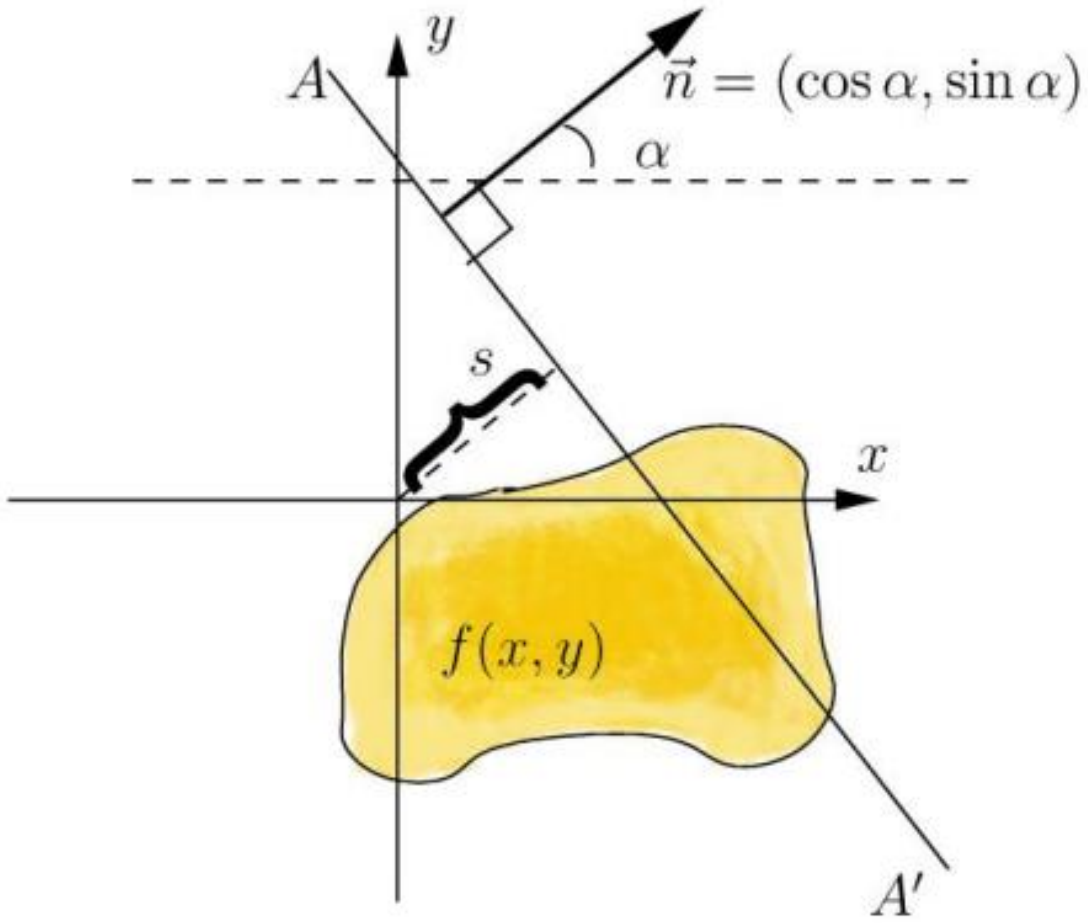


Рисунок 1.2 – Двовимірне перетворення Радона

Пізніше було виявлено, що краще використовувати дисперсію замість суми значень пікселів уздовж лінії проекції . Дисперсія набагато краще обробляє розриви яскравості вздовж лінії проекції. Такі розриви з'являються через границі, які ортогональні лінії проекції.

Тепер визначимо ПВД. Нехай *Г(α)* – набір пікселів на лінії проекції, яка відповідає цьому куту. Нехай *(x',y')* - координати центрального пікселя на зображені. Піксель *(x,y)* ϵ *Г(α)* тоді і тільки тоді, коли:

$$-\frac{1}{2} \leq (x - x') \cdot \cos\alpha + (y - y') \sin\alpha \leq \frac{1}{2}.$$

Визначення 1.3 (ПВД, radial variance vector):

Нехай *I(x,y)* означає яскравість пікселя *(x,y)*, *#Г(α)* – потужність безлічі, тоді ПВД *R[α]*, де α = 0.1,…, 179 визначимо як:

$$R[\alpha] = \frac{\sum_{(x,y)\in\Gamma(\alpha)} I^2(x,y)}{\#\Gamma(\alpha)} - \left(\frac{\sum_{(x,y)\in\Gamma(\alpha)} I^2(x,y)}{\#\Gamma(\alpha)}\right)^2. \quad (1.6)$$

Досить побудувати вектор по 180 значенням кута, так як перетворення Радону є симетричним. Отриманий вектор можна використовувати для побудови хешу, але в цьому алгоритмі пропонується ще деяке поліпшення: застосувати ДКП до отриманого вектору. В результаті отримаємо вектор, що успадковує всі важливі властивості ДКП. Для хешу беруться перші сорок коефіцієнтів отриманого вектора, які відповідають низьким частотам. Таким чином, розмір отриманого хешу - 40 байт.

Отже, розглянемо кроки побудови хешу за допомогою даного алгоритму.

1. Прибрати колір. Для усунення непотрібних високих частот;
2. Розмити зображення (blurring) за допомогою використання Гаусова розмиття [15]. Зображення перетвориться за допомогою функції Гауса. Для усунення деяких шумів;
3. Застосувати гамма-корекцію. Для усунення бляклості зображення.
4. Побудувати променевий вектор дисперсії;
5. Застосувати ДКП до вектору дисперсії;
6. Побудувати хеш.

1.4 Хеш-функція на основі оператора Марра-Хілдрета

Оператор Марра-Хілдрет [16,17] дозволяє визначати границі на зображенні. Взагалі кажучи, границю на зображенні можна визначити як край або контур, що відокремлює сусідні частини зображення, які мають порівняно відмінні характеристики відповідно до деяких особливостей. Цими особливостями можуть

бути колір або текстура, але частіше за все використовують сіру градацію кольору зображення (яскравість).

Результатом визначення меж є карта кордонів. Карта кордонів описує класифікацію меж для кожного пікселя зображення. Якщо границю визначати як різку зміну яскравості, то для їх знаходження можна використовувати похідні або градієнт.

Нехай функція $f_c(x)$ позначає рівень яскравості для лінії (одновимірний масив пікселів). Перший підхід визначення меж полягає в знаходженні локальних екстремумів функції, тобто перших похідних. Другий підхід (метод Лапласа) полягає в розрахунку других похідних $f_c(x)$.

Обидва підходи можуть бути адаптовані для випадку двовимірних дискретних зображень, але з деякими проблемами. для знаходження похідних в дискретному випадку потрібно застосувати апроксимацію.

Крім того, шум на зображенні може значно погіршити процес пошуку кордонів. Тому перед визначенням меж до зображення потрібно застосувати будь-якої фільтр, що пригнічує шуми. Для побудови хешу був обраний алгоритм, який використовує оператор Лапласа (2 підхід) і фільтр Гауса.

Визначення 1.4 (Безперервний Лапласіан, оператор Лапласа).

Нехай $f_c(x)$ визначає яскравість на зображенні. тоді безперервний Лапласіан визначимо як:

$$\nabla^2 f_c(x,y) = \nabla \cdot \nabla f_c(x,y) = \frac{d^2 f_c(x,y)}{dx^2} + \frac{d^2 f_c(x,y)}{dy^2}. \quad (1.7)$$

Перетворенні в нуль $\nabla f_c(x,y)$ являються точками, що відповідають границі функції $f_c(x,y)$, так як це точки, при яких друга похідна наближається до нуля. Різні фільтри (дискретні оператори Лапласа) можуть бути отримані з безперервного Лапласіан. Такий фільтр, $h(n_1,n_2)$, може бути застосований до дискретного зображення за допомогою використання згортки функцій. Оператор Лапласа для зображення $f(n_1,n_2)$ можна переписати як:

$$\nabla^2 f(n_1, n_2) = f(n_1, n_2) * h(n_1, n_2), \quad (1.8)$$

де * - згортка функцій.

Щоб побудувати карту меж, потрібно знайти звернення в нуль дискретного оператора $\nabla^2 f(n_1, n_2)$.

Тепер розглянемо оператор Марра-Хілдрет. Він також називається Лапласіан від фільтра Гауса (далі LoG) - це особливий тип дискретного оператора Лапласа.

LoG конструюється за допомогою застосування оператора Лапласа до фільтру (функції) Гауса. особливість цього оператора в тому, що він може виділяти границі в певному масштабі. Змінну масштабу можна варіювати для того, щоб краще виявити границі.

Визначення 1.5 (Фільтр Гауса).

Фільтр Гауса визначений як:

$$g_c(x, y) = e^{\frac{x^2+y^2}{2\sigma^2}}, \quad (1.9)$$

де σ - стандартне відхилення розподілу Гаусса;

x - відстань від початку координат по горизонтальній осі;

y - відстань від початку координат по вертикальній осі.

Згортку з операцією Лапласа можна поміняти місцями, тому що похідна і згортка - лінійні оператори:

$$\nabla^2[f_c(x, y) * g_c(x, y)] = [\nabla^2 g_c(x, y)] * f_c(x, y). \quad (1.10)$$

Ця властивість дозволяє заздалегідь обчислити оператор $\nabla^2 g_c(x, y)$, тому що він ніяк не залежить від зображення $f_c(x, y)$.

Визначення 1.6 (оператор Марра-Хілдрет, LoG).

LoG $h_c(x, y)$ визначимо як:

$$h_c(x,y) = \Delta^2 g_c(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4} \cdot e^{\frac{x^2+y^2}{2\sigma^2}}. \quad (1.11)$$

Щоб використовувати LoG в дискретній формі, зробимо дискретизацію даного рівняння, підставивши потрібну змінну масштабу. За стандартом її значення приймається як 1.0. Потім фільтр можна застосувати до зображення, використовуючи дискретну згортку.

Визначення 1.7 (Згортка з бібліотеки CImg).

Нехай x, y, z - піксельна ширина, довжина і глибина зображення I. Результат R згортки зображення I с маскою M визначимо як:

$$R(x,y,z) = \sum_{i,j,k} I(x-i, y-j, z-k) M(i,j,k). \quad (1.12)$$

Тепер розглянемо етапи алгоритму, що використовує для побудови хешу оператор Марра-Хілдрет.

1. Прибрати колір. Для усунення непотрібних високих частот;
2. Зменшити зображення до розміру 128×128;
3. Розмити зображення (blurring) за допомогою розмиття Гауса (рис.1.3(a)). Зображення обробляється за допомогою функції Гауса. Для усунення деяких шумів;
4. Побудувати оператор Марра-Хілдрет;
5. Застосувати дискретну згортку до LoG і зображення. Вийде зображення, на якому чітко видно скачки яскравості (рис.1.3(b)).
6. Перетворити зображення в гістограму. Зображення розбивається на маленькі блоки (5×5), в яких підсумовуються значення яскравості (рис.1.3(c)).
7. Побудувати хеш з гістограми. Гістограма розбивається на блоки 3x3. Для цих блоків вважається середнє значення яскравості і використовується метод побудови ланцюжка бітів, в результаті отримуємо бінарний хеш розміром 64 байта.

a) Розмиття Гауса b) Застосування LoG c)Гістограма яскравості

Рисунок 1.3 – Приклад застосування алгоритма Marr-Hildreth Operator Based Hash до зображення

Розмір отриманого хешу не маленький, проте, порівняння двох хешів займає досить незначний час в порівнянні з Radial алгоритмом. Також такий алгоритм чутливий до поворотів зображення, але стійкий до масштабування, затемнення, стиску.

Вище представлені методи стійкі до різного роду атак на зображення, що, у свою чергу, вирішує проблему методу хеш-стеганографії.

В результаті аналізу та тестування алгоритмів було запропоновано модифікувати метод хеш-стеганографії за допомогою перцептивного хеш-алгоритму Discrete Cosine Transform Based Hash. Таким чином стійкість хеш-коду до змін структури зображення підвищується. Тобто якщо зловмисник, що перехопив повідомлення, спотворить якість зображення, то це ніяк не вплине на хеш-код його. Також причинами вибору стали легкість в реалізації та адаптації даного алгоритму під структуру нашого методу.

2 МОДИФІКАЦІЯ МЕТОДУ ХЕШ-СТЕГАНОГРАФІЇ

2.1 Основні відомості про хеш-стеганографію

Стеганографія сьогодні розрахована на те, що передане повідомлення шифрується спочатку досить стійким криптографічним кодом[21]. Потім дане повідомлення вбудовується в контейнер (зображення). Як правило при цьому в зображенні з'являються шуми або зміни, непомітні звичайній людині. І в більшості випадків і для шифрування і дешифрування потрібен ключ, який теж необхідно передати будь - яким чином, або при зустрічі, або по закритому каналу зв'язку, який, зазвичай, відсутня.

Стеганографія - це мистецтво і наука про приховування даних в обкладинках, що невинно виглядають, щоб ніхто не міг виявити саме існування прихованих даних.

Це дещо відрізняється від криптографії, так як мета стеганографії - скритність, а не тільки секретність [22]. Наприклад, зашифрований текст може містити особливі слова, такі як «QJYZQDFLKJ», але стеганоповідомлення (текстовий файл з вбудованими даними) повинен виглядати як звичайний текстовий файл, щоб не викликати підозр у секретному повідомленні.

Сама стеганографія має довгу історію, і поширення цифрової комунікації підвищило важливість комп'ютерної стеганографії, за допомогою якої можна приховати сам канал цифрового зв'язку.

Програма шифрування електронної пошти може захистити вміст повідомлення, але вона не може приховати сам факт передачі повідомлення. Якщо Ви відправите зашифрований лист своїй дівчині у ворожу країну, скоріше за все Ви можете бути заарештовані як підозрюваний у шпигунстві, якщо ви не дасте поліцейським прочитати вміст.

З безпечною системою стеганографії, яка приховує протокол передачі пошти в іншому протоколі, ви можете відправляти зашифровані листи, не побоюючись викликати таких підозр. Цей тип застосування систем стеганографії є більш серйозним в областях, де використання криптографії обмежено законом.

Деякі країни (наприклад, в Сполучених Штатах, Франції, Росії) забороняють необмежене використання надійної криптографії і зберігають за собою право прослуховування телефонних розмов з дозволу суду.

Дехто каже, що економічно ефективна можливість прослуховування телефонних розмов запобігає злочинній діяльності. Інші кажуть, що це коштує дорого і може зробити всю систему вразливою. Важливість стеганографії з'являється тут. Якщо люди, чиї повідомлення прослуховуються, можуть використовувати безпечну систему стеганографії, ефект від прослуховування, дозволеного судом, зменшиться.

Якби безпечна стеганографічна система буде легкою у використанні і може забезпечити достатню пропускну здатність, ефект від прослуховування був би незначним. Приховане спілкування зі стеганографічними системами може бути заборонено, але буде дуже важко (або неможливо) його виявити і пред'явити позов. Більшість злочинців використовуватимуть систему безпечної стеганографії для злочинної діяльності, і тільки кримінальні повідомлення будуть приватними. Отже, коли і як можна захистити стеганографічна система, важливо вимірювати економічну ефективність політик обмеження криптографії, таких як політика умовного депонування ключів.

Як відомо для захисту переданих даних використовується криптографія, але наявність зашифрованого повідомлення привертає до себе увагу і створює інтерес до злому переданого повідомлення.

Для того, щоб цього уникнути використовують не чисту криптографію, а криптографію спільно зі стеганографії. Стеганографія приховує сам факт передачі інформації. Наприклад, передача повідомлення відбувається під виглядом передачі JPEG - зображення. Однак існуючі методи стеганографії можуть так само залучати зловмисників, так як можуть змінювати і спотворювати зображення.

Зловмисники можуть знати про такі методи, і цілком можливо, що зображення з секретною інформацією вони зможуть відрізнити від звичайних, з подальшим розкриттям цієї інформації. Іншими словами, існуючі методи

стеганографії можуть змінювати зображення, шляхом вставки в нього контейнера з інформацією.

На сьогоднішній день в стеганографії умовно виділяють декілька напрямків:

а) класична стеганографія, включає в себе «некомп'ютерні методи» приховування повідомлення неелектричної природи;

б) комп'ютерна стеганографія, передбачає використання властивостей форматів даних, що оброблюються та передаються в інформаційно-комунікаційних мережах.

в) цифрова стеганографія, заснована на надмірності мультимедійних даних, що передаються, поданих у цифровому вигляді, які мають аналогову природу(зображення, відео, звук) з самого початку.

Хеш-стеганографія [18,23] взагалі нічого не вбудовує в стегоконтейнер. Якщо нічого не вбудовуємо, то порожній контейнер не відрізняється від стегоконтейнера.

Ідея хеш-стеганографії достатньо легка і зрозуміла кожному.

Беремо велику кількість зображень. В сучасному світі надлишку гаджетів, дешевої пам'яті для зберігання фотографій тощо, генерація великої кількості зображень не складе особливих труднощів. Також для генерації зображень можна використати спеціальні генератори на основі нейронних мереж, які можуть створювати нові зображення.

Якщо не має бажання генерувати, існує безліч сайтів, що надають можливість загрузити на комп'ютер величезну кількість різноманітних зображень.

Наступна задача полягає у виборі хеш-функції. Проблем з вибором хеш-функції виникнути не повинно.

Щоб звузити коло пошуків обираємо головним критерієм рівноймовірність, тому можемо обрати (з криптографічної точки зору), наприклад, MD5.

Алгоритм функціонування даного методу представлений нижче.

1. Як було сказано вище, нам потрібна велика бібліотека зображень. Використовуємо ресурси інтернету або генеруємо зображення за допомогою

спеціальних програм. Єдиною умовою для усіх зображень являється їх унікальність.

2. Обчислюємо значення хеш-функції для кожного зображення. Обираємо перші(або останні) n нібблів (напівбайт) з хеш-коду (рис.2.1).

Рисунок 2.1 – Перші символи хеш-кодів деяких зображень

3. Для прискорення роботи з бібліотекою слід записати словник зображення – хеш-функція у файл, для більш зручного пошуку потрібного хеш-коду [20].

4. Переводимо повідомлення у шістнадцятирічну систему числення, наприклад, 68656c69636f70746572. Розбиваємо його на n нібблів. У якості прикладу обираємо n=2, тоді потрібно розбити наше повідомлення по 2 ніббла. Якщо «розбивати» будемо пробілами, то виглядати це буде так: 68 65 6c 69 63 6f 70 74 65 72. Отримані «шматочки», кожен з яких містить n нібблів будемо називати словами. Тобто всього у нас вийшло 10 слів. Наступні кроки описують пошук ідентичних зображень.

4.1. Беремо «слово». Шукаємо зображення у якій хеш-код збігається з нашим словом. Якщо таких зображень кілька, беремо будь-яку з них.

4.2. Передаємо зображення в канал.

4.3. Беремо друге «слово», шукаємо хеш-код що збігається з другим словом.

4.4. Передаємо зображення в канал.

4.5. Зі «словами», що залишилися, проводимо ті ж самі операції.

В результаті отримаємо набір зображень, які відповідають нашому повідомленню.

Слід зазначити, якщо взяти довільним чином зображення в базі даних, то ймовірність того, що її хеш-код буде збігатися з довільно заданим словом дорівнює $\frac{1}{2^{4n}}$. Для n = 1 ця величина дорівнює $\frac{1}{16}$, для n = 2 - $\frac{1}{256}$. Чим менша ця ймовірність, тим більше зображень необхідно помістити в базу даних для задоволення наших стеганографічних потреб.

З іншого боку, чим більше n, тим більше швидкість коду. На жаль зі збільшенням n, швидкість збільшується лінійно, а потреба в кількості фотографій експоненціально.

Формальне пояснення стегосистеми виглядає наступним чином.

1. Визначимо стегосистему як безліч об'єктів *I*.
2. Визначимо безліч слів *S*.
3. Визначимо функцію *H(i)*, яке для кожного *i* з *I* ставить в відповідність слово *s* з *S*. Назвемо цю функцію хеш-кодом.
4. Для передачі повідомлення *s1, s2, s3, ..., sm* слід знайти такі *i1, i2, i3, ..., im*, для яких справедливо: *H (i1) = s1, H (i2) = s2, ..., H (im) = sm.*

2.2 Перші згадки про алгоритм хеш-стеганографії

В 1998 році Кашен (Cachin) запропонував теоретико-інформаційний підхід до стеганографії [22], в рамках якого, зокрема, була визначена так названа досконала стегосистема, в якій повідомлення, що несуть і не несуть приховану інформацію, статистично невиразні.

Також там описана універсальна стеганографічна система, для якої ця властивість виконується тільки асимптотично, при збільшені довжини

повідомлення, до того ж важкість кодування та декодування виростає експоненціально.

За визначенням система універсальна, якщо вона може бути застосована і в тому випадку, коли імовірнісні характеристики повідомлень, які використовують для передачі прихованої інформації, відомі не повністю.

Для того, щоб розглядати цю тему в її найбільш загальному вигляді, узагальнюємо базову модель стегосистеми (рис.2.2), наступним чином.

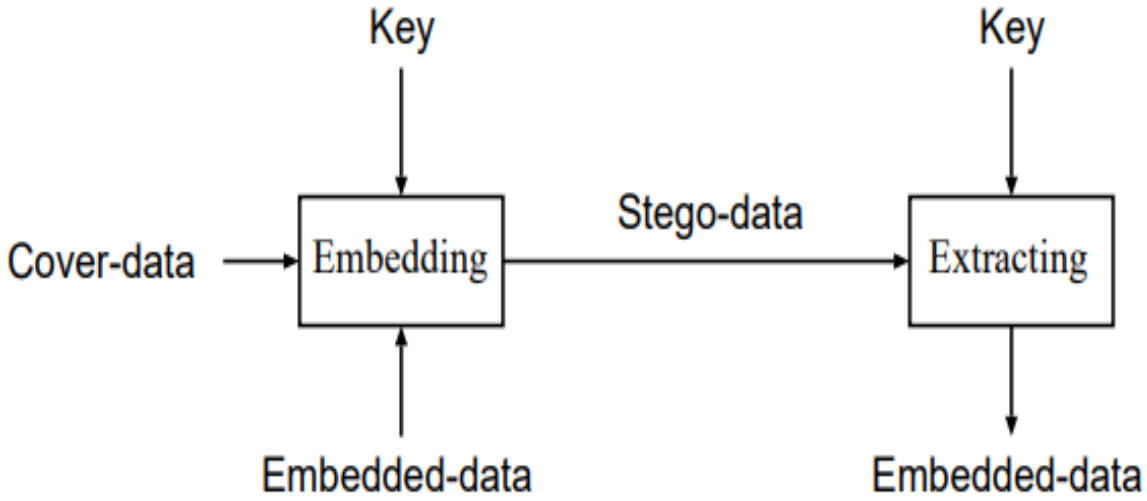


Рисунок 2.2 – Базова модель стегосистеми по Кашену

1. Відправник може використовувати внесок навколишнього середовища, відмінний від даних обкладинки та вбудованих даних. Наприклад, деяка інформація про користувача поштою незамінна для створення або налаштування вмісту пошти алгоритмічно без введення неприродності. Стегосистеми, які використовують метод вибору, вимагають декількох кандидатів на покриття даних, які також можуть розглядатися як додатковий внесок середовища до алгоритму вбудовування.

2. Результати одержувача повідомлення не повинні бути ідентичними вихідному повідомленню. Іншими словами, стегосистема може зробити помилку в цій моделі

Окрім базової моделі стегосистеми існує також удосконалена. Удосконалена модель зображена на рисунку 2.3 і складається з таких елементів, як :

а) вбудовування $\mathcal{E}$;

б) вилучення $\mathcal{D}$;

в) випадкові величини, які формують входи і виходи $\mathcal{E}$ і $\mathcal{D}$.

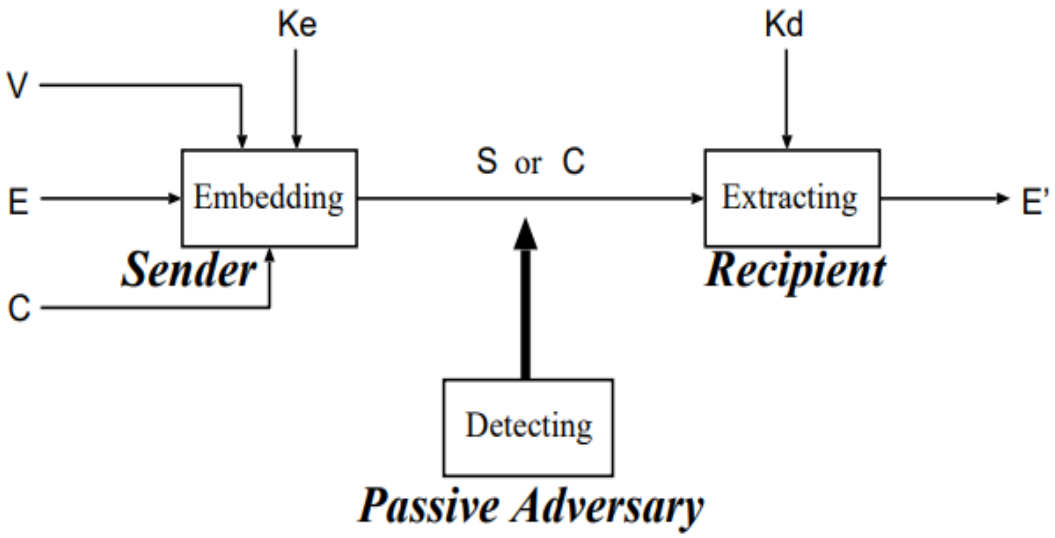


Рисунок 2.3 – Схема одноразової хеш-стеганографії

Опишемо процес вбудовування як:

$$S = \mathcal{E}(E, C, K, V), \tag{2.2}$$

де $E \epsilon$ **Е –** вбудовані дані;

$C \epsilon$ **М –** дані зображення;

$V \epsilon$ **V** – зовнішні дані;

$K \epsilon$ **К** – загальний секретний ключ;

$S \epsilon$ **М** – стегодані.

Аналогічно, позначимо процес вилучення:

$$E' = \mathcal{D}(S, K), \tag{2.3}$$

де E' є **E** – наближення до оригіналу E.

Мета пасивного зловмисника полягає в тому, щоб визначити, чи використовує відправник стегосистему чи ні, прослуховуючи повідомлення (S або C) [16]. Повідомлення, яке пасивний зловмисник підслуховує, - це реалізація або S (у першому випадку), або C (у останньому випадку). Чим більше розподіл S стає подібним до розподілу C, тим зловмиснику важче виявити використання стегосистеми.

У крайньому випадку обидва розподіли стають однаковими, а підслуховування стає безглуздим. Цей тип стегосистеми називають цілком безпечним.

Для будь-якого пасивного зловмисника будь-якої обчислювальної потужності неможливо виявити використання абсолютно безпечної стегосистеми.

Тепер зробимо деякі додаткові припущення, щоб зробити модель придатною для комп'ютерних стегосистем і виключити деякі абсурдні ситуації:

а) вхідні та вихідні дані повинні бути обчислені алгоритмами;

б) відправник та одержувач можуть використовувати приватні генератори випадкових чисел;

в) всі специфікації стегосистеми (такі як алгоритми генератора ключів, вбудовування і вилучення) відомі противникам. Іншими словами, безпека стегосистеми не залежить від секретності його алгоритмів. Усі розділені секрети знаходяться в ключі;

г) секретний ключ K повинен генеруватися і розподілятися між відправником і одержувачем надійно перед передачею стего-даних;

2.3 Модифікація методу хеш-стеганографії

В результаті аналізу було вирішено модифікувати метод хеш-стеганографії наступним чином :

- інтегрувати перцептивний хеш-алгоритм заснованого на дискретному косинусному перетворені для отримання стійкого хеш-коду зображення;

- кодувати текст за допомогою таблиці кодування символів.

Хеш-алгоритм виглядає наступним чином. Зменшуємо розмір зображення до 32×32. Переходимо до зображення в градаціях сірого. Обчислюємо ДКП. Для роботи нам потрібні низькі частоти, як було зазначено вище, тому в даному алгоритмі використовують коефіцієнти ДКП верхнього лівого блоку 8×8. Вони відповідають найнижчим частотам зображення. Обчислюємо середнє значення коефіцієнтів ДКП за виключенням першого коефіцієнта, оскільки він може значно відрізнятися від інших значень і буде відкидати середнє значення. Поставимо у відповідність коефіцієнтам ДКП бінарну послідовність, де 0 відповідає коефіцієнтам, значення яких менше за середнє, 1 – більше. Переводимо 64-бітову послідовність у 64-бітове ціле число.

Таким чином модифікований метод хеш-стеганографії буде виглядає наступним чином.

Перш за все необхідно створити базу даних (далі БД) хеш-кодів зображень. Для цього нам знадобиться велика кількість унікальних зображень, тому скачуємо їх з Інтернету. Бажано мати у наявності якомога більше ніж тисяча, даної кількості досить для коректної роботи. Далі до кожного зображення ми застосовуємо вище вказаний хеш-алгоритм, в результаті чого ми отримуємо унікальні хеш-коді(рис 2.4).

```
2d0a3f8d5771159aa0550d8303f1e653
b9849cfb669347efd6d60cafcde1ceca
f8abba0c8ff2ccfad0e0c286bc528345
5cf50d5dc9738df496510a51325b2465
163d5e1683b5cdabd9f8210fd5e03c52
a8c3c72c145493fdf85ad278506995ae
40e1cf6642858ca40de1610856f2feba
e88cfbc77f1387a6d84545421b6fa02f
c957353d4918dd02b75c7efdf5a8f6bf
48ab13e219db91145d11d91a1fabc64b
80b7d19e75ed47908f221d01abd44b92
d43b777d7fda07e2f09a99ded4b30eca
4f89f53988d828242eb2022518ba9003
ac00ff94be4ba410775d59600c9ca674
```

Рисунок 2.4 – Хеш-коди зображень

Отримані хеш-коди будуть використовуватись при шифруванні повідомлення. Користувач вводить повідомлення, воно кодується згідно ASCII таблиці символів.

Після чого для кожної пари символів ASCII-коду у БД відбираються зображення, у яких перші два символи хеш-коду співпадають із хеш-кодом повідомлення(рис 2.5).

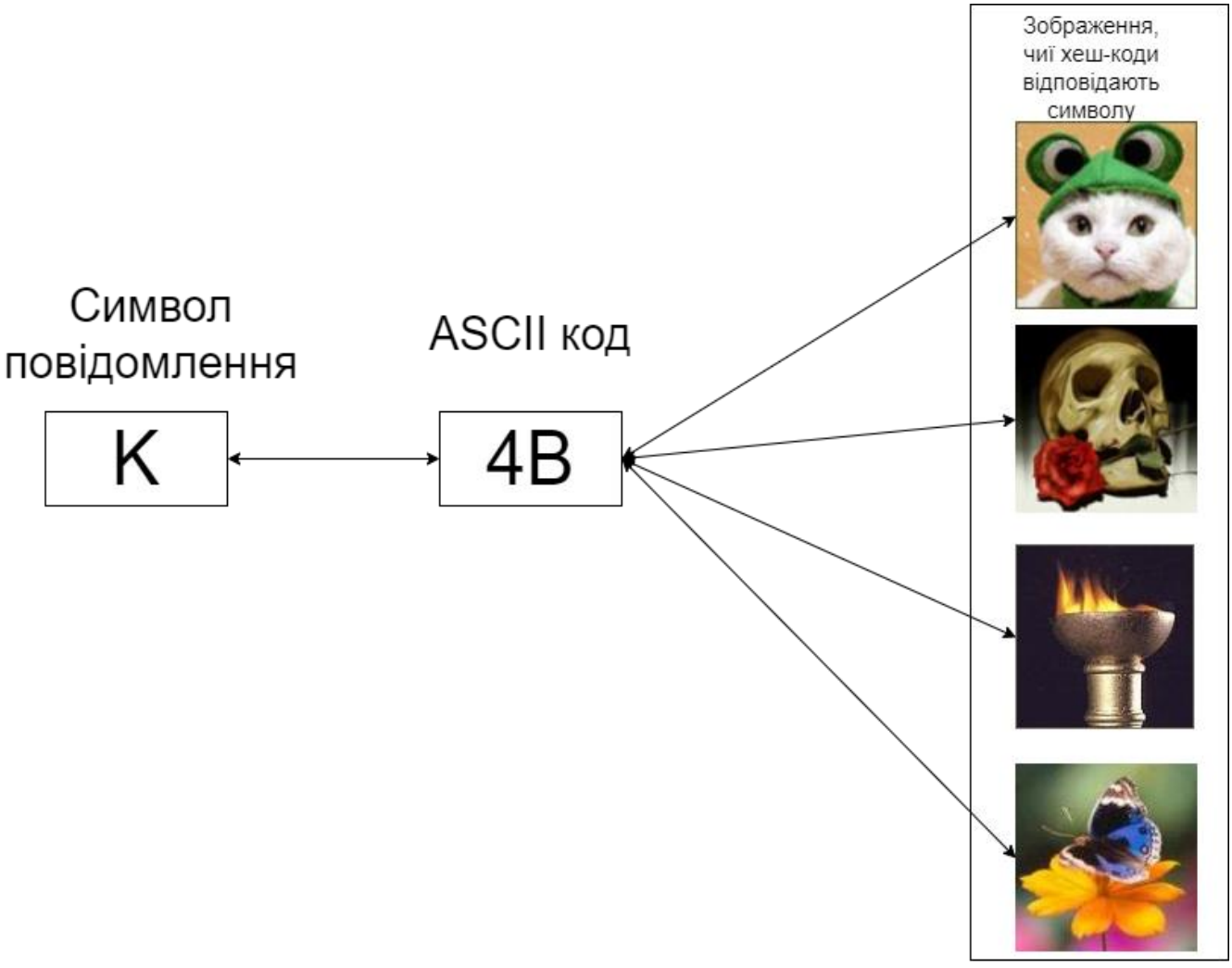


Рисунок 2.5 – Схема відповідності «символ-зображення»

Для кожного символу повідомлення підбирається унікальне зображення, тобто якщо один і той же символ зустрічається у тексті більше ніж один раз, то для кожного з них буде підібране інше зображення відмінно від попередніх (рис.2.6)

(якщо не можливо підібрати для другого символу відмінне зображення, то береться попереднє використане).

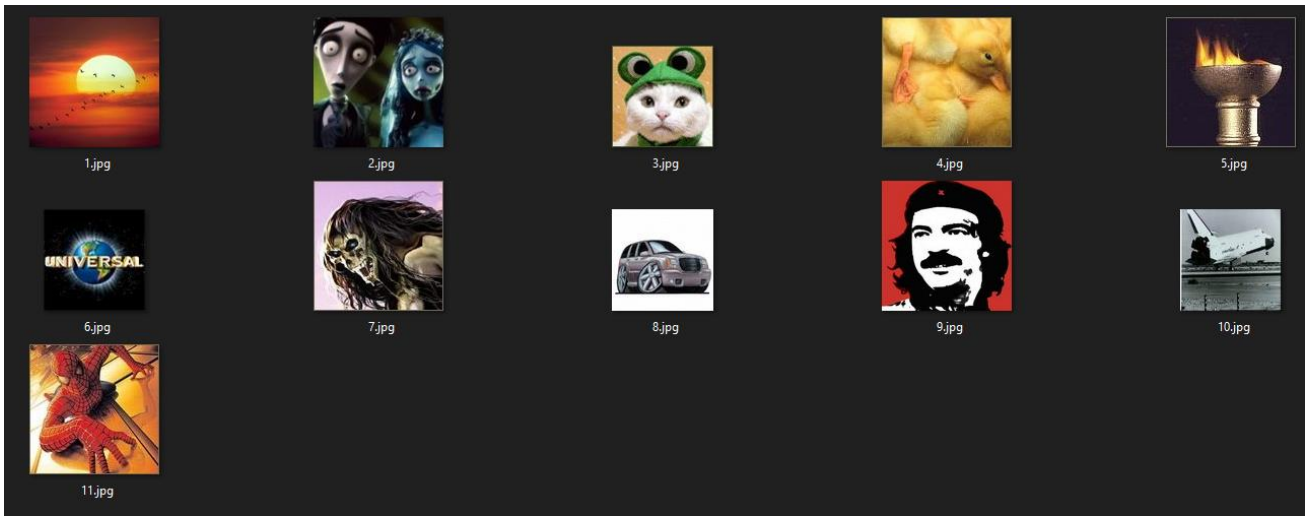


Рисунок 2.6 – Відібрані зображення, які шифрують словосполучення «Bad company»

Перцептивні хеш-коди - це інша концепція в порівнянні з криптографічними хеш-функціями на зразок MD5[24] і SHA1.[25] У криптографії кожен хеш-код є випадковим. Дані, які використовуються для генерації хеша, виконують роль джерела випадкових чисел, так що однакові дані дадуть однаковий результат, а різні дані - різний результат.

У порівнянні двох хеш-кодів MD5 насправді можна зробити тільки два висновки. Якщо хеші відрізняються, значить, дані різні. Якщо хеші збігаються, то і дані, швидше за все, однакові (оскільки існує ймовірність колізій, то однакові хеш-коди не гарантують збігу даних). На відміну від них, перцептивні хеші можна порівнювати між собою і робити висновок про ступінь відмінності двох наборів даних.

Наприклад, при передачі зображення зловмисник вирішив провести атаку на якість зображення, з ціллю пошкодити дані. Розрахувавши хеш-код зображень за допомогою MD5, то, після атаки, відновити оригінальне повідомлення буде не можливо. Можемо зробити висновок, що такий алгоритм являється не стійким до атак.

Використовуючи замість MD5 перцептивний хеш-алгоритм, хеш-код набуває стійкості до таких видів атак, як:

а) масштабування;

б) стиск;

в) розтягнення;

г) зміна яскравості або контрасту;

д) маніпуляції з кольорами.

Щодо кодування самого повідомлення, то тут критерієм є лише зручність та ефективність у використанні. Таблиць кодування символів існує вдосталь, загальноприйнятими являються таблиці кодування ASCII та UNICODE.

ASCII і Unicode - це два стандарти кодування в електронній комунікації. Вони використовуються для представлення тексту в комп'ютерах, в телекомунікаційних пристроях та іншому обладнанні.

По суті, вони є стандартами того, як представляти різницеві символи у двійковому форматі, щоб їх можна було записувати, зберігати, передавати і зчитувати на цифрових носіях.

Основна відмінність між ASCII і Unicode полягає в тому, що ASCII представляє малі літери (a-z), великі літери (A-Z), цифри (0-9) і символи, такі як знаки пунктуації, тоді як Unicode представляє літери англійської, арабської, грецької тощо, математичні символи, історичні сценарії та «смайлики», що охоплюють широкий діапазон символів, ніж ASCII.

Спочатку ASCII використовував сім бітів для кодування кожного символу. Пізніше це число було збільшено до восьми з розширеним ASCII для усунення очевидної неадекватності оригіналу.

Unicode, у свою чергу, використовує програму кодування змінних бітів, де ви можете вибирати між 32, 16 і 8-бітними кодуваннями. Використання більшої кількості бітів дозволяє використовувати більше символів за рахунок великих файлів, в той час як менша кількість бітів дає вам обмежений вибір, але ви економите багато місця. Використання меншої кількості бітів (тобто UTF-8 або ASCII), ймовірно, було б краще, якщо ви кодуєте великий документ англійською

мовою. Для даної роботи було вирішено використовувати таблицю кодування ASCII.

У другому розділі детально розглянуто алгоритм хеш-стеганографії та наведені його недоліки.

Для вирішення проблеми була вирішено модифікувати метод хеш-стеганографії за допомогою хеш-алгоритму на основі Perceptual(DCT) Hash. Також для симетричного шифрування повідомлення було вирішено використовувати таблицю кодування символі ASCII.

В результаті модифікації алгоритм краще забезпечує стійкість стеганоповідомлення до різних видів атак, а також до стеганоаналізу.

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

3.1 Середовище програмування

Для розробки програмного продукту обрано мову об’єктно-орієнтовного програмування C# (Visual Studio 2017). Дана мова використовує об'єктно-орієнтований підхід до програмування у всьому. Це означає, що потрібно буде описувати абстрактні конструкції на основі предметної області, а потім реалізовувати між ними взаємодія. Даний підхід користується великою популярністю, тому що дозволяє не тримати в голові всю інформацію, а працювати за принципом чорного ящика: подав вхідні дані.

C# - одна з найпопулярніших мова програмування. Це важливо для розробників, оскільки популярність мови прямо пропорційна тому, наскільки для нього будуть доступні онлайн-матеріали.

Головна причина вибору C# криється у величезній кількості інструментів і фреймворків, які підтримує ця мова. Майже будь-який розробник з готовністю визнає, що Visual Studio є однією з найбільш багатофункціональних і потужних середовищ розробки на ринку. Фреймворк .Net надає сотні бібліотек для створення веб-додатків, забезпечення безпеки, роботи з файловими системами і т.д.

C# в значній мірі підтримується Microsoft. Нові функції та синтаксичні поліпшення з'являються набагато швидше, ніж в інших мовах програмування, як, наприклад, в Java.

Гнучкість мови C# є величезною перевагою, в порівнянні з деякими мовами програмування. Різноманітність додатків, які можуть бути розроблені за допомогою C#, .Net і Visual Studio, має широкий спектр можливостей:

а) додатки для Windows;

б) мобільні додатки;

в) веб-додатки;

г) додатки для Android і iOS, які розробляються за допомогою додаткових фреймворків , таких як Xamarin або Mono.

Звичайно, всі ці речі можливо виконувати і за допомогою інших мов програмування, але зазвичай, в таких випадках, використовуються сторонні інструменти інших розробників. Програмісти, які працюють з C# мають згуртований набір інструментів, які підтримуються Microsoft для розробки будь-якого типу програми.

C# має ряд переваг, такі як:

а) можливість розширення системи (в C# можна спокійно довантажувати будь-які .exe-файли, імпортувати класи і об'єкти з інших програм).

б) кросплатформеність (mono, концепція NET).

в) складність розробки і супроводу (підбір кадрів, читаність коду, документованість мови).

г) ступінь відкритості вихідних текстів бібліотек, виконуваних програм, кількість літератури і підтримку (MSDN).

д) можливість залучення сторонніх розробників при розробці системи для програмування вузько-спеціалізованих завдань (складання, модулі, ті самі exe).

е) захищеність і контроль версій алгоритмів, які підключаються (концепція NET).

ж) трудомісткість написання (той же NET).

з) швидкість роботи (Розподіл процесів, розподіл даних швидкість роботи з даними).

и) зручність розробки.

3.2 Розробка програмного продукту

Розробка програмного продукту починається реалізації перцептивного хеш-алгоритму. Після завантаження зображень в систему, кожне зображення підлягає обробці для отримання індивідуального хеш-коду.

Зображення зменшується до розмірів 32x32 та переводиться в градації сірого.

Наступним кроком потрібно розрахувати дискретне косинусне перетворення. Для наглядності код представлений нижче.

Далі розраховуємо середнє значення , але тільки для верхнього лівого блоку 8×8 матриці коефіцієнтів дискретного косинусного. Кожен піксель зображення замінюється на 0 або 1 в залежності від того, більше він або менше середнього значення. В результаті отримаємо зображення з значення пікселів якого дорівнює 0 або 1 (рис.3.1).

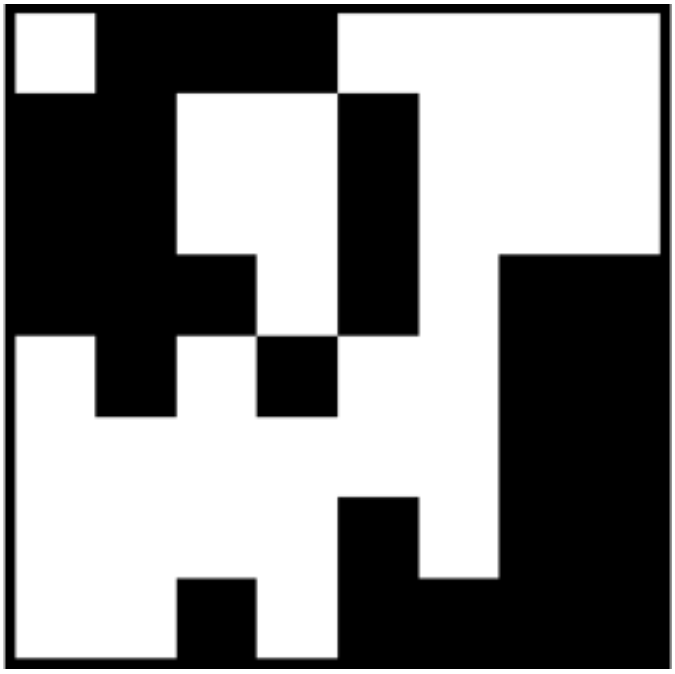


Рисунок 3.1 – Приклад зображення у бітовому вигляді

Будуємо ланцюжок бітів, тобто записуємо біти в певній послідовності, а саме зліва направо та зверху вниз. Отримане значення кодуємо за допомогою хеш-функції MD5, результатом буде хеш-код зображення (рис. 3.2).

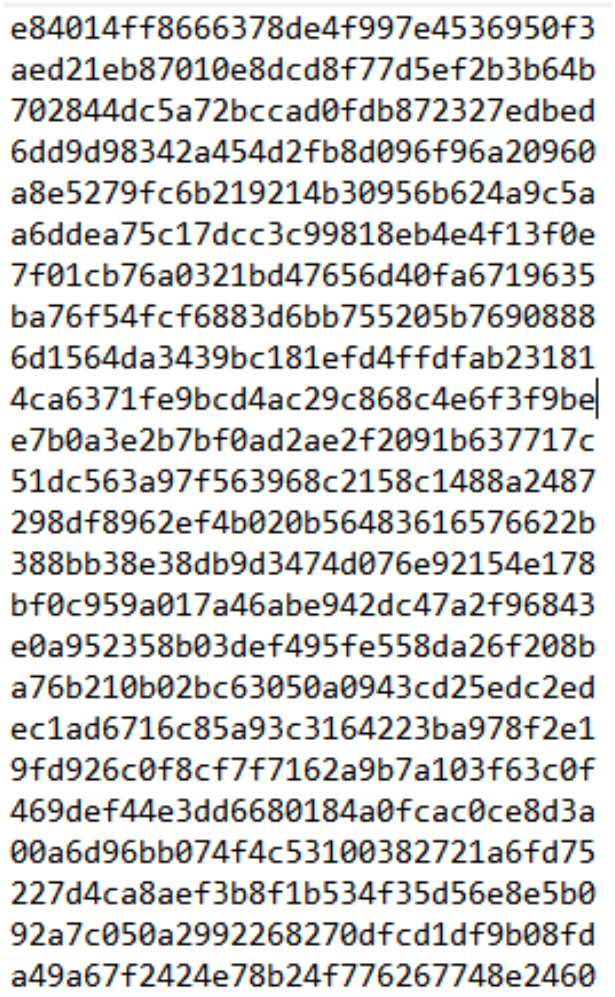


Рисунок 3.2 – База даних хеш-кодів зображень

Таким чином розраховується хеш-код для кожного зображення і для зручності у подальшому використанні записуються в .txt файл.

Наступний етап розробки безпосередньо алгоритму кодування повідомлення. Введене користувачем повідомлення зчитується у систему та шифрується за допомогою таблиці кодування символів ASCII (рис. 3.3). Отриманий в результаті шифрування хеш-код записуємо у масив по два символи для подальшого кодування.

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Рисунок 3.3 – Кодування символів за допомогою ASCII

Потім на кожен елемент масиву потрібно знайти зображення, у якого ідентичний хеш-код. Беремо перший елемент і шукаємо зображення у якого хеш-код збігається з нашим словом. Підходяще зображення копіюється у нову папку та перейменовується (рис. 3.4).

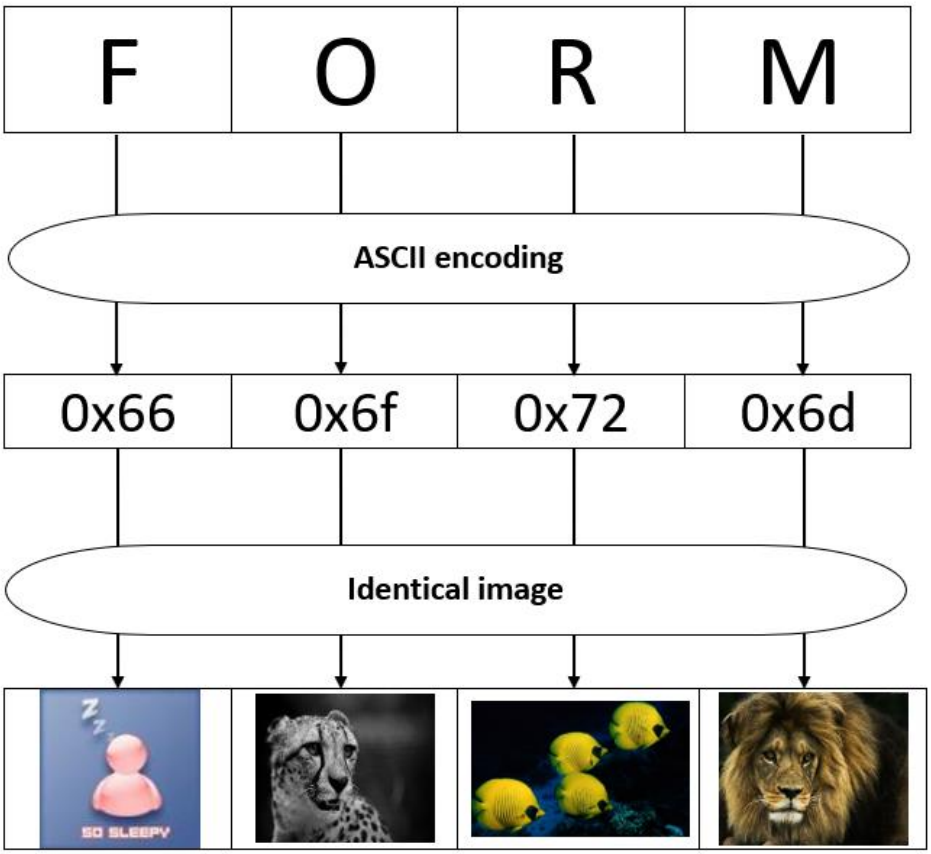


Рисунок 3.4 –Кодування повідомлення

Наступний етап – декодування стеганоповідомлення. Користувач вказує шлях до папки, в якій зберігається стеганоповідомлення. Програма зчитує зображення у систему для подальшої роботи.

За допомогою хеш-алгоритму визначається хеш-код зображень. Виписуємо з хеш-кодів перші два символи, отримуємо бітову послідовність. За допомогою таблиці кодування символів ASCII декодуємо повідомлення та виводимо на екран (рис 3.5).

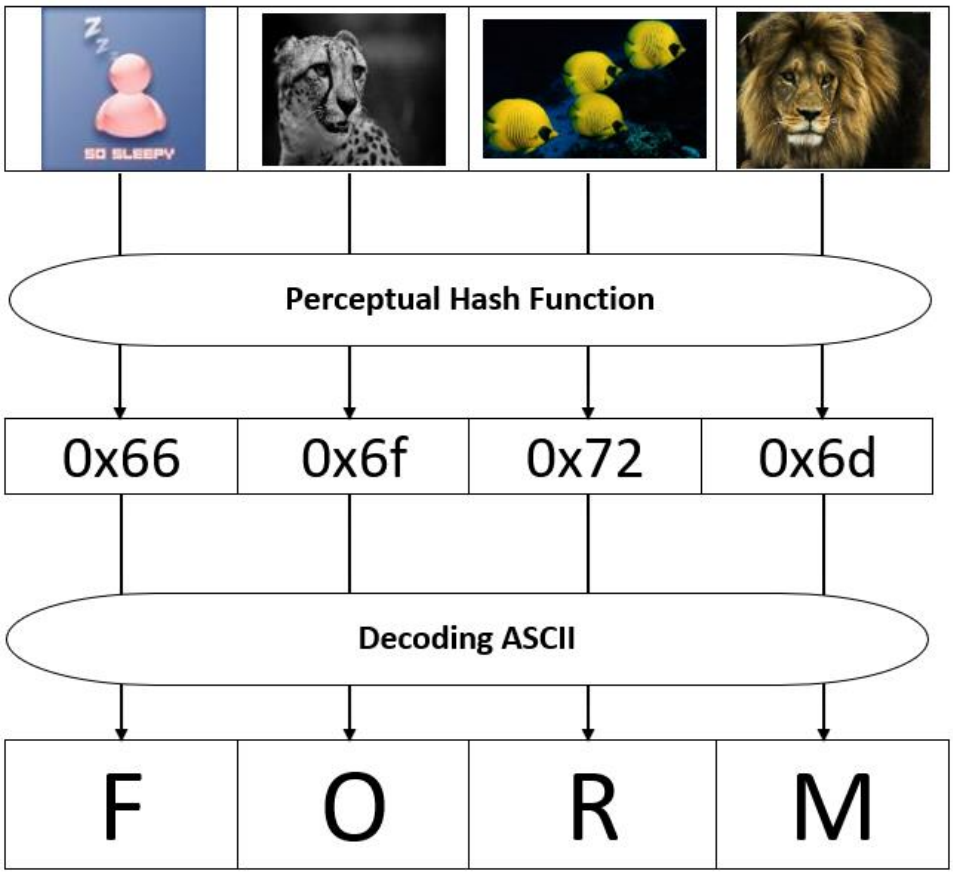


Рисунок 3.5 –Декодування стеганоповідомлення

Кінцевий етап - реалізація функції очищення полів. Коли користувач закінчив роботу з кодуванням/декодуванням повідомлення, він натискає кнопку «Clear Text», текст в полях для введення та виведення. Після чого користувач може продовжувати роботу с програмою.

Детально ознайомитись з кодом програмного продукту можна у додатку А Код програмного продукту.

3.3 Програмний інтерфейс

Для розробки програмного інтерфейсу була використана платформа .NET з технологією Windows From [18]. Інтерфейс користувача (UI) - це сукупність інформаційної моделі проблемної області, засобів і способів взаємодії користувача з інформаційною моделлю, а також компонентів, що забезпечують формування інформаційної моделі в процесі роботи програмної системи. Людині, що користується даним середовище, не потрібно створювати скрипт або вводити команди командного рядка для виконання завдань. На відміну від програм кодування для виконання завдань, користувачу не потрібно розуміти деталі виконання завдань.

Компоненти такого інтерфейсу можуть містити меню, панелі інструментів, кнопки, перемикачі, списки та повзунки – і це лише деякі з них.

Інтерфейс даної програми включає в себе такі елементи управління як: 5 функціональних кнопок (Encode, Decode, Load Folder, Clear Text, Update Base), 2 спливаючих меню, 1 текстове поле вводу та 1 текстове поле виводу. Інтерфейс користувача зображено на рисунку 3.7.

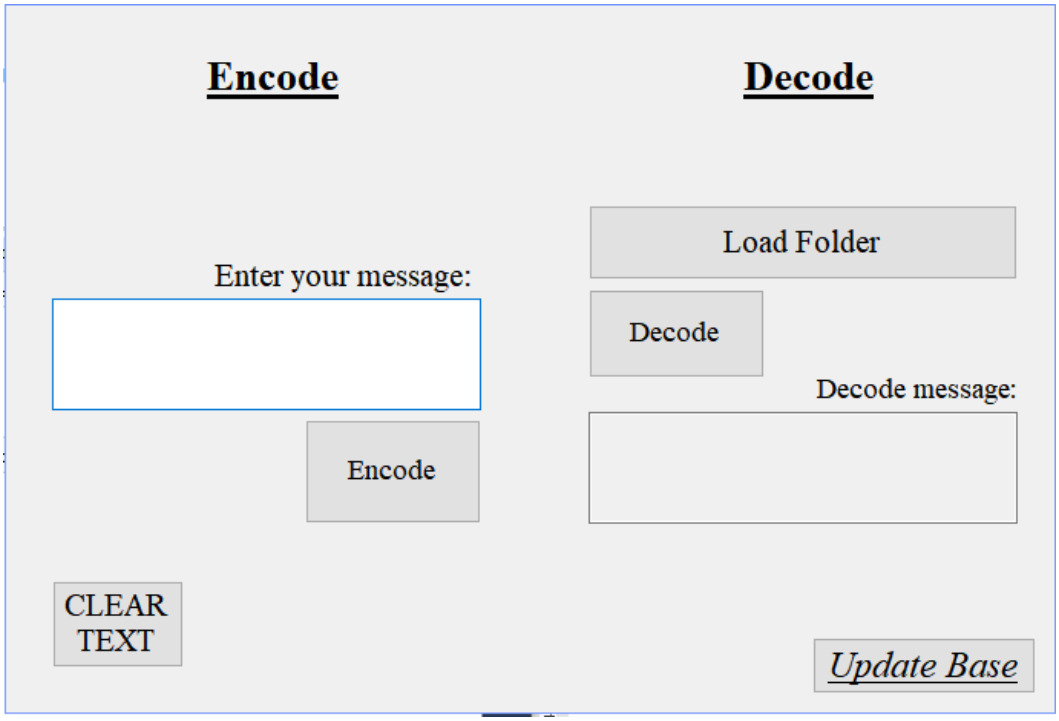


Рисунок 3.7 – Програмний інтерфейс користувача

Перед початком роботи з програмою необхідно створити або оновити існуючий список хеш-кодів зображень.

Для цього користувачеві потрібно натиснути на кнопку «Update Base», після чого відкриється додаткове вікно, у якому потрібно вказати шлях до папки із зображеннями, які будуть використовуватися при шифруванні повідомлень(рис. 3.8).

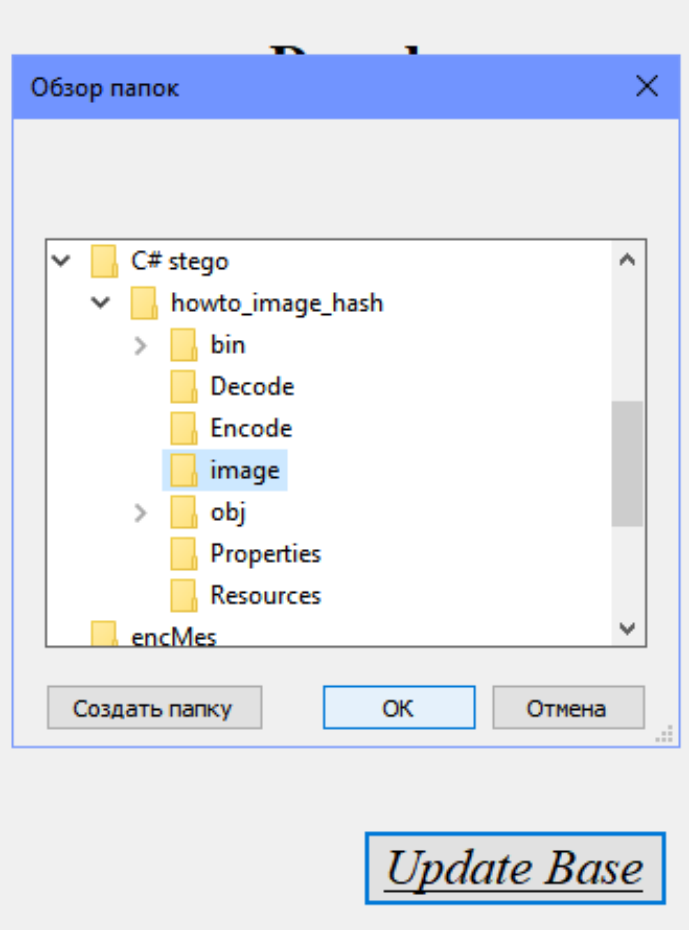


Рисунок 3.8 – Оновлення списку хеш-кодів

Процес оновлення займе деякий час, тривалість залежить від кількості зображень.

Для шифрування необхідно в полі «Enter your message» ввести повідомлення, після чого натиснути кнопку «Encode». Програма кодує повідомлення і відкриває папку із зображеннями, які представляють оригінальне повідомлення (рис. 3.9).

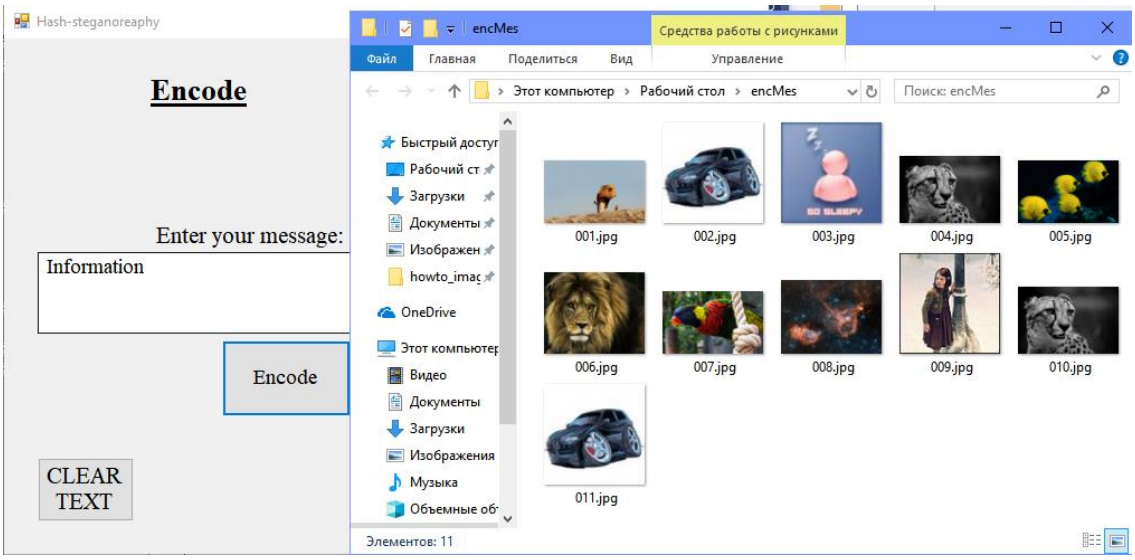


Рисунок 3.9 – Закодоване повідомлення

Для декодування потрібно натиснути кнопку «Load Folder», одразу відкриється додаткове вікно, де потрібно обрати папку, в якій знаходяться зображення стеганоповідомлення (рис. 3.10).

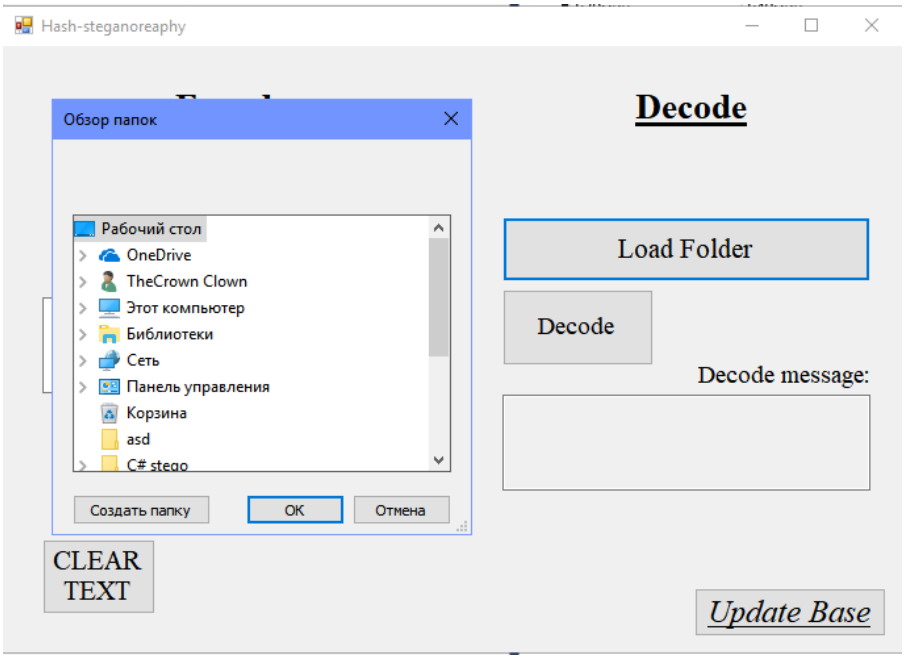


Рисунок 3.10 – Завантаження папки з зображеннями для декодування

Після чого натискаємо на кнопку «Decode» і в текстовому вікні нижче буде виведене зашифроване повідомлення (рис. 3.11).

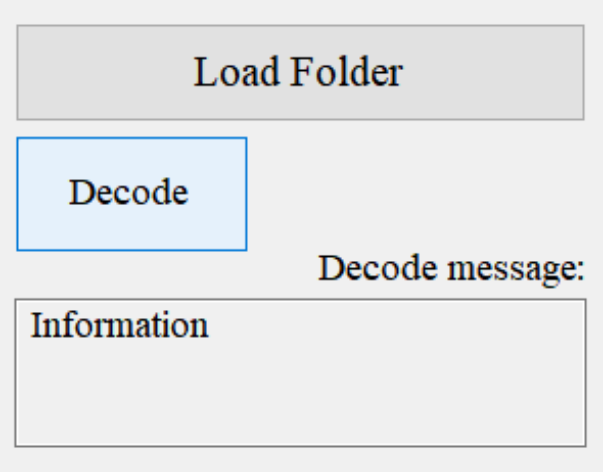


Рисунок 3.11 – Декодоване повідомлення

Після закінчення шифрування/дешифрування рекомендується натиснути кнопку «Clear Text», щоб очистити текстові поля введення/виведення для подальшої роботи.

У даному розділі детально розглянули процес розробки програмного продукту та інтерфейсу користувача, описано причини вибору середовища програмування.

Інтерфейс зроблений максимально зручним і легким як у використанні, так і розумінні, жодних зайвих елементів. Код програмного продукту розділений на блоки, що безпосередньо полегшує читабельність та розуміння порядку виконання процесів прописаних у ньому.

ВИСНОВОК

В ході виконання кваліфікаційної роботи проведено аналіз сучасних алгоритмів хеш-стеганографії та існуючих перцептивних хеш-алгоритмів для обчислення хеш-коду зображень.

За основу для модифікації методу хеш-стеганографії обрано перцептивний хеш-алгоритм заснований на дискретному косинусному перетворені.

Модифікований алгоритм хеш-стеганографії дозволяє підвищити стійкість хеш-кодів зображень до різноманітних атак на зображення – масштабування, стиснення, корекція яскравості та контрастності зображення, а також захистити від стеганоаналізу.

Алгоритм реалізовано на мові об’єктно-орієнтовного програмування C# (Visual Studio 2019). За допомогою Windows Form сконструйовано легкий та зрозумілий інтерфейс користувача.

Мету досягнуто, усі поставлені в роботі задачі вирішено.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бохонько М.В., В.В. Зоріло ,О.Ю. Лебедєва. Модифікація методу хеш-стеганографії, заснованого на передачі послідовності цифрових зображень. *Інформатика та математичні методи в моделюванні*. 2020р. С.605-608.

2. Денисюк В. О. Стеганографічний алгоритм захисту даних з використанням файлів зображень. *Ефективна економіка*. 2017. V.5. URL: http://www.economy.nayka.com.ua/?op=1&z=5584

3. Н.Е.Губенко ,Д.С. Сипаков. Анализ особенностей методов цифровой стеганографии для защиты информации, передаваемой по открытым каналам. *Информатика и кибернетика*. 2015. С.28-38

4. Choosing and Optimizing Cryptographic Algorithms for Resource-Constrained Systems. URL: https://www.sciencedirect.com/topics/computer-science/hash-algorithm

5. Simple and DCT perceptual hash-algorithms [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://www.hackerfactor.com/blog/index.php?/archives/432-Looks-Like-It.html.

6. Egmont-Petersen, M., de Ridder, D., Handels, H. Image processing with neural networks - a review // Pattern Recognition – V. 35 (10) - P. 2279–2301.

7. Zeng Jie. A Novel Block-DCT and PCA Based Image Perceptual Hashing Algorithm // IJCSI International Journal of Computer Science Issues – 2013. - V.10. – P. 399-403.

8. Standaert, F.X., Lefebvre, F., Rouvroy, G., Macq, B.M., Quisquater, J.J., and Legat, J.D. Practical evaluation of a radial soft hash algorithm // In Proceedings of the International Symposium on Information Technology: Coding and Computing (ITCC) - V. 2 - P. 89-94.

9. D. Marrand, E. Hildret. Theory of edge detection // Proc. R. Soc. Lond - 1980. – V.207 – P. 187-215

10. Advances in Imaging and Electron Physics. URL: https://www.sciencedirect.com/topics/mathematics/discrete-cosine-transform

11. Discrete cosine transform [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://en.wikipedia.org/wiki/Discrete_cosine_transform

12. Median filter [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://en.wikipedia.org/wiki/Median_filter

13. Median [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://en.wikipedia.org/wiki/Median

14. France A. The Radon transform and its inverse. 2011. URL: https://www.researchgate.net/publication/267230532_The_Radon_transform_and_its_inverse

15. Jen Beatty. The Radon Transform and the Mathematics of Medical Imaging. Honors Theses. 2012. URL: https://digitalcommons.colby.edu/honorstheses/646/

16. Gaussian blur [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://en.wikipedia.org/wiki/Gaussian_blur

17. Marr-Hildreth-Operator [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http://de.wikipedia.org/wiki/Marr-Hildreth-Operator

18. Juan Miguel Valverde. Laplacian and Marr-Hildreth operator.2015. URL: http://laid.delanover.com/second-order-edge-detection-operators-laplacian-and-marr-hildreth-operator/

19. Знакомьтесь: Хеш-стеганография. Очень медленная, но совершенно секретная [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: https://habr.com/ru/post/272935/

20. Генне О.В. Основні положення стеганографії // Защита информации. Конфидент №3 (2000) [Електронний ресурс]. - Режим доступу: World Wide Web. - URL: http: // citforum.ru / internet / securities / stegano.shtml (дата звернення 31.05.2019)

21. Gruhl, D. Bender W. Information hiding to foil the casual counterfeiter // LNCS, 1998. - V. 1525. – P. 1-15.

22. Anita Pradhan , K. Raja Sekhar, Gandharba Swain. Digital Image Steganography Using LSB Substitution, PVD, and EMD. *Mathematical problems in Engineering.* 2018. URL: https://www.hindawi.com/journals/mpe/2018/1804953/

23. Cachin, C. An information-theoretic model for steganography // LNCS, 1998. - V. 1525. – P. 306–318.

24. Shin N. One-Time Hash Steganography // Lecture Notes in Computer Science, 2000. – V. 1768. – P. 17-28

25. K. Krishna Prasad ,P. S. Aithal. A Study on Fingerprint Hash Code Generation Based on MD5 Algorithm and Freeman Chain Code. *International Journal of Computational Research and Development*. 2018. V.3. P.13-22.

26. Tim Grembowski, Roar Lien, Kris Gaj, Nghi Nguyen,Peter Bellows, Jaroslav Flidr, Tom Lehman, Brian Schott. Comparative Analysisofthe HardwareImplementationsofHashFunctionsSHA-1andSHA-512. *Electrical and Computer Engineering*. 2002. URL: https://www.academia.edu/9367680/Comparative_Analysis_of_the_Hardware_Implementations_of_Hash_Functions_SHA1_and_SHA512

27. Руководство по программированию в Windows Forms URL: World Wide Web. - URL: https://metanit.com/sharp/windowsforms/

28. Березуцький, В. В. Навч. пос. Т. С. Бондаренко, Г. Г. Валенко та ін. / за заг. ред. В. В. Березуцького. — 2-ге вид., перероб. і доп. — Х.: Факт, 2007. — 480 с.

29. Охорона праці в офісі. Вимоги до робочого місця офісного працівника. URL: https://gc.ua/uk/oxorona-praci-v-ofisi-vimogi-do-robochogo-miscya-ofisnogo-pracivnika/

30. Жидецький В.Ц., Джигирей В.С., Мельник О.В. Основи охорони праці. Львів: Афіша, 2001. 350 с

31. Кодексом Законів «Про працю» (ст. 14. Обов’язки працівника щодо додержання вимог нормативно-правових актів з охорони праці)

32. ДСанПІН 3.3.2.007-98. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. [Чинний від 1998.12.10]. Київ, 1998.

33. ДБН В.2.5-28:2018. Природне і штучне освітлення URL: https://dnaop.com/html/2032/doc-ДБН_В.2.5-28-2006

34. ДСТУ EN 12464-1:2016 Світло та освітлення. Освітлення робочих місць URL: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=71838

35. ДСН 3.3.6.042-99. Державні санітарні норми мікроклімату виробничих приміщень. [Чинний від 1999.12.01]. Київ, 1999. 63 с.

36. СН 4088-86. Санітарні норми мікроклімату виробничих приміщень . URL: https://files.stroyinf.ru/Data2/1/4293786/4293786335.htm

37. ДНАОП 0.03-3.06-80 Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень. URL: https://dnaop.com/html/2296/doc-ГН_2152-80

38. ДСанПІН 3.3.2.007-98. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин. [Чинний від 1998.12.10]. Київ, 1998.

39. ДСН 3.3.6.037-99. Санітарні норми виробничого шуму, ультразвуку та інфразвуку URL: https://dnaop.com/html/40957/doc-ДСН_3.3.6.037-99

40. ГН 2152-80 (ДНАОП 0.03-3.06-80) Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень URL:https://otipb.at.ua/load/gn_2152_80_dnaop_0_03_3_06_80_sanitarno_gigienichni_normi_dopustimikh_rivniv_ionizaciji_povitrja_virobnichikh_ta_gromadskikh_primishhen/2-1-0-2350

41. НПАОП 40.1-1.32-01. Правила будови електроустановок. Електрообладнання спеціальних установок. URL: https://dnaop.com/html/1692/doc-НПАОП_40.1-1.32-01

42. ДСТУ 13109-97. Електрична енергія. Сумісність технічних засобів. Норми якості електричної енергії в системах електропостачання загального призначення. URL: http://www.loe.ant.lviv.ua/home/dokumenti/gost-dstu

43. ДСН 3.3.6.042-99. Санитарные нормы микроклимата производственных помещений. URL: https://dnaop.com/html/31678/doc ДСН_3.3.6.042-99

44. НАПБ А.01.001-2014 Правила пожежної безпеки в Україні URL: http://online.budstandart.com/ua/catalog/doc-page?id_doc=60541

45. ДСТУ 8829:2019. Пожежовибухонебезпечність речовин і матеріалів URL: https://ammokote.com/wp-content/uploads/2020/08/dstu_8829_2019.pdf

46. ДСТУ Б В.1.1-11:2005. Електричні кабельні лінії. Метод випробування на вогнестійкість. URL: https://dnaop.com/html/43872/doc ДСТУ_Б_В.1.1-11_2005

47. ОНТП 24-86. Визначення категорій приміщень і будинків по вибухопожежної і пожежної небезпеки URL: https://dnaop.com/html/2590/doc-ОНТП_24-86

48. ДСТУ 8828:2019 Пожежна безпека. Загальні положення. URL: http://online.budstandart.com/ua/catalog/doc-page.html?id_doc=82138

49. ДСТУ ISO 3864-1:2005. Графические символы. Цвета и знаки безопасности. Часть 1. Принципы проектирования знаков безопасности для рабочих мест и мест общественного назначения. (ISO 3864-1:2002, IDT) http://online.budstandart.com/ru/catalog/doc-page?id_doc=50319

50. ДСТУ ISO 6309:2007. Протипожежний захист. URL: http://fire.berdyansk.net/help/znaki_iso_6309-2007/dstu_iso_6309-2007_znaky_bezpeky.htm