1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. **Институт компьютерных наук и кибербезопасности**

**ЛАБОРАТОРНАЯ РАБОТА № 8**

1. «Основы cтеганографической защиты информации»
2. по дисциплине «Основы информационной безопасности»
3. Выполнил
4. студент гр. 5131001/30003 Шевчук Н.Е.

1. Руководитель
2. асс. преподавателя Орел Е.М.
4. Санкт-Петербург

2024

Оглавление

[1. Цель работы 3](#_Toc163482416)

[2. Формулировка задания 3](#_Toc163482417)

[3. Теоретическая справка 3](#_Toc163482418)

[4. Ход работы 5](#_Toc163482419)

[5. Результаты работы 9](#_Toc163482420)

[6. Вывод 12](#_Toc163482421)

[7. Контрольные вопросы 12](#_Toc163482422)

[8. Приложение 16](#_Toc163482423)

# 1. Цель работы

Приобретение навыков исследования свойств стегоконтейнеров, разработки стегосистем и их применения для сокрытия данных при передаче с помощью графических сообщений.

# 2. Формулировка задания

* Создать программу-кодер, реализующую стеганографическое сокрытие “секретного” текстового файла в созданный графический файл;
* Сравнить исходный графический файл формата bmp, созданный заранее в Paint, с изображением, полученным в ходе работы программы-кодера и содержащим упакованное “секретное” сообщение;
* Создать программу-декодер, осуществляющую извлечение “секретного” сообщения из стегоконтейнера.

# 3. Теоретическая справка

Основные технологии, используемые для сокрытия информации, — это

криптография и стеганография. Криптография применяется для шифрования, то есть применения защиты данных путем криптографических алгоритмов.

Стеганография позволяет скрывать сам факт наличия тайного сообщения. При

этом скрываемое сообщение встраивается в некоторый не привлекающий

внимания объект-контейнер, который может открыто пересылаться адресату.

Развитие средств вычислительной техники дало толчок развитию

компьютерной стеганографии, когда сообщение встраивают в различные

цифровые данные: графические изображения, аудио-и видеозаписи.

Цифровая стеганография включает в себя следующие направления:

• встраивание цифровых водяных знаков (ЦВЗ) (watermarking);

• встраивание идентификационных номеров (fingerprinting);

• встраивание заголовков (саptioning);

• встраивание информации с целью ее скрытой передачи.

Для интеллектуальной собственности, представленной в цифровом виде,

актуальна проблема защиты авторских прав. Данная проблема решается с

помощью встраивания в защищаемый объект невидимых меток ЦВЗ. Такие знаки могут содержать некоторый аутентичный код, например, информацию о собственнике или авторе. Невидимые ЦВЗ анализируются специальным

декодером, проверяющим их аутентичность.

Технология встраивания идентификационных номеров производителей

имеет много общего с технологией ЦВЗ. Отличие заключается в том, что при

встраивании номеров защищенная копия имеет свой уникальный идентификатор.

Встраивание невидимых заголовков служит для каталогизации и хранения

разнородно представленной информации как единого целого.

Встраивание информации с целью ее скрытой передачи является

старейшим способом применения стеганографии.

**Стегоконтейнер** – информационная последовательность, в которой

скрывается сообщение.

**Предварительный кодер** – устройство, предназначенное для преобразования скрываемого сообщения к виду, удобному для встраивания в стегоконтейнер.

**Кодер** - устройство, предназначенное для осуществления вложения

сообщения в другие данные с учетом их модели, например, путем модификации младших значащих бит. Например, в изображении с 256 градациями серого (каждый пиксель изображения кодируется 8 битами), глаз человека не способен заметить различия в цвете на один разряд, поэтому младший бит используют для сокрытия информации. Данный метод прост в реализации и эффективен.

**Детектор-** устройство, предназначенное для определения наличия

стегосообщения.

**Декодер** - устройство, восстанавливающее скрытое сообщение. Данный

узел может отсутствовать.

Стегосистема образует стегоканал, по которому передается заполненный

стегоконтейнер. Данный канал считается подверженным воздействиям со

стороны нарушителей или помех.

# 4. Ход работы

В данной лабораторной работе используется изображение в качестве стегоконтейнера. Файл изображения имеет формат BMP (Bitmap). Заголовок файла имеет следующие поля, которые занимают 14 байт:

Таблица 1 – Заголовок файла BMP

|  |  |
| --- | --- |
| Длина поля | Описание поля |
| 2 | Код формата, в данном случае «BM» |
| 4 | Размер файла в байтах |
| 2 | 0 (резервное поле) |
| 2 | 0 (резервное поле) |
| 4 | Смещение от начала до растровых данных |

Таблица 2 – Заголовок BITMAP (Информация об изображении):

|  |  |
| --- | --- |
| *Длина поля* | Описание поля |
| *4* | Размер заголовка BITMAP в байтах (40) |
| *4* | Ширина изображения в пикселях |
| *4* | Высота изображения в пикселях |
| *2* | Число плоскостей (не используется) = 1 |
| *2* | Бит/пиксель\* |
| *4* | Тип сжатия (обычно не используется) |
| *4* | Размер сжатого изображения в байтах |
| *4* | Горизонтальное расширение, пиксель/м |
| *4* | Вертикальное расширение, пиксель/м |
| *4* | Кол-во используемых цветов (очень редко) |
| *4* | Кол-во "важных" цветов (не используется) |

Глубина изображения файлов BMP в данной работе 24bit. Битовая глубина относится к количеству битов, используемых для представления цвета пикселя. Эти 24 бита разделена на три канала по 8 бит, которые представляют красный, зеленый и синий соответственно. Таким образов, в изображении может быть использовано различных цветов. При такой глубине изображения палитра отсутствует.

В данной лабораторной программе реализован метод замены наименее значимых бит изображения на биты сообщения. Поскольку замене подвергаются лишь наименее значимые биты, разница между исходным изображением-контейнером и контейнером, содержащим скрытые данные, невелика и обычно незаметна для человеческого глаза. Количество заменяемых бит называется степенью упаковки. В зависимости от нее можно вложить различный размер сообщения.

Рисунок 1 – Исходное изображение

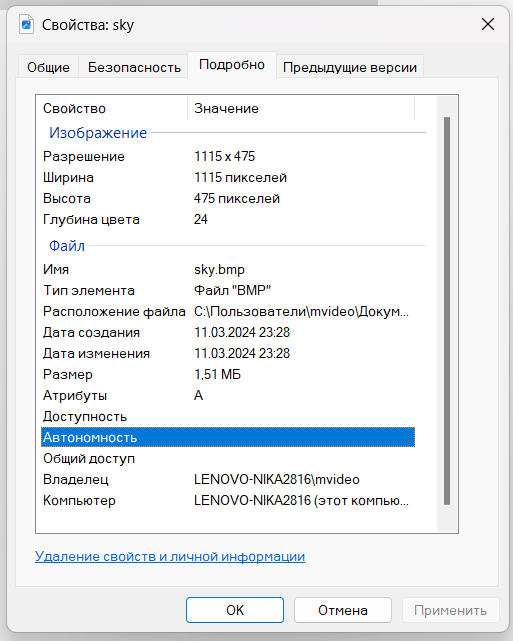


Рисунок 2 – Свойства исходного изображения

Параметры данного изображения. Размер файла составляет 1,51 МБ. Глубина цвета: 24. Ширина составляет 1115 пикселей, а длина – 475 пикселей.

Для вычисления содержательной части данного графического файла потребуются вышеуказанные параметры. Подчитаем ее размер:

байт.

Далее вычислим максимальную длину скрываемого сообщения, помещаемого в созданный стегоконтейнер: (54 байта мы сохраняем практически без изменений чтобы исходная картинка и картинка с сообщением имели одинаковые конфигурационные параметры)

байт.

Далее реализована программа кодер и закодировано некоторое

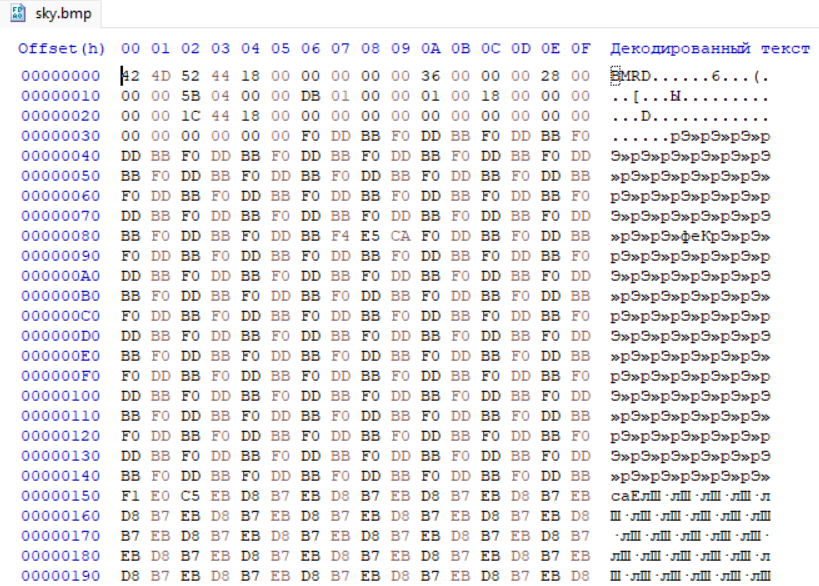
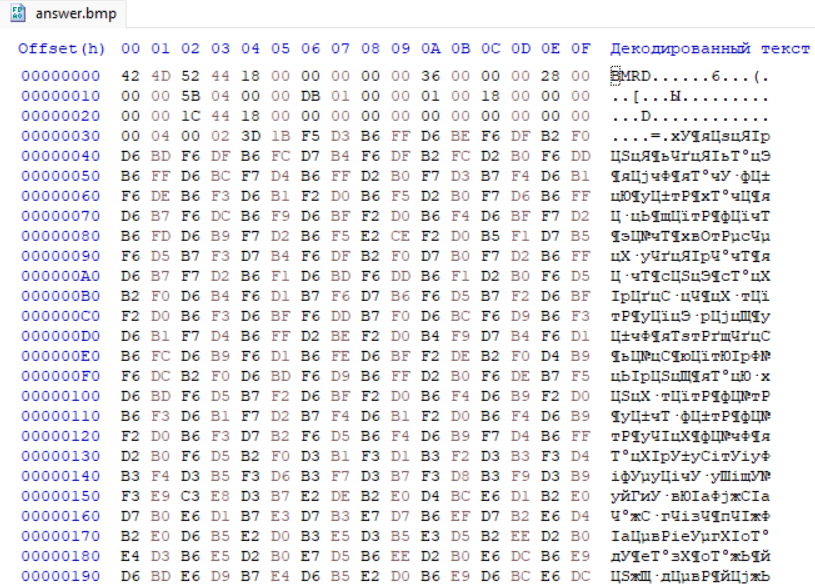
сообщение. Используя программу HxD при сравнении исходного файла и файла с вложенным сообщением, можно заметить, код изображения изменился. Но размеры файлов одинаковые, «на глаз» изображения одинаковые. Это происходит, потому что биты меняются незначительно, а также не добавляются новые биты.

Рисунок 3 – Сравнение кодов изображения до и после кодирования сообщения

Можно наблюдать, что изменения в коде произошли, но при этом размер

изображения не изменился, так как мы не добавляли биты/байты, а только

заменяли имеющиеся. Заметим, что при кодировании сообщения мы вложили в неиспользуемый 49-ый байт информацию о степени упаковки, в 50-54 байт длин стегосообщения для дальнейшей возможности декодирования.

# 5. Результаты работы

Реализовав программу кодер мы проводим тесты с различными степенями упаковки, для большей наглядности проведем их с большим текстом:

При степени упаковки 1 изменений не видно невооруженным взяглядом.



Рисунок 4 – Степень упаковки 1

При степени упаковки 2 изменений также не видно.

Рисунок 5 – Степень упаковки 2

При степени упаковки 4 изменения незначительны.

Рисунок 6 – степень упаковки 4

Степень упаковки равна 8, пиксели имеют совершенно другой цвет. Появилась внизу полоска серого цвета.

Рисунок 7 – степень упаковки 8

При попытке закодировать слишком большой текст, программа выводит уведомление о невозможности такого действия. Реализовав программу декодер и сравнив ее результат с изначальным текстовым файлом, можно убедиться, что обе программы работает правильно.

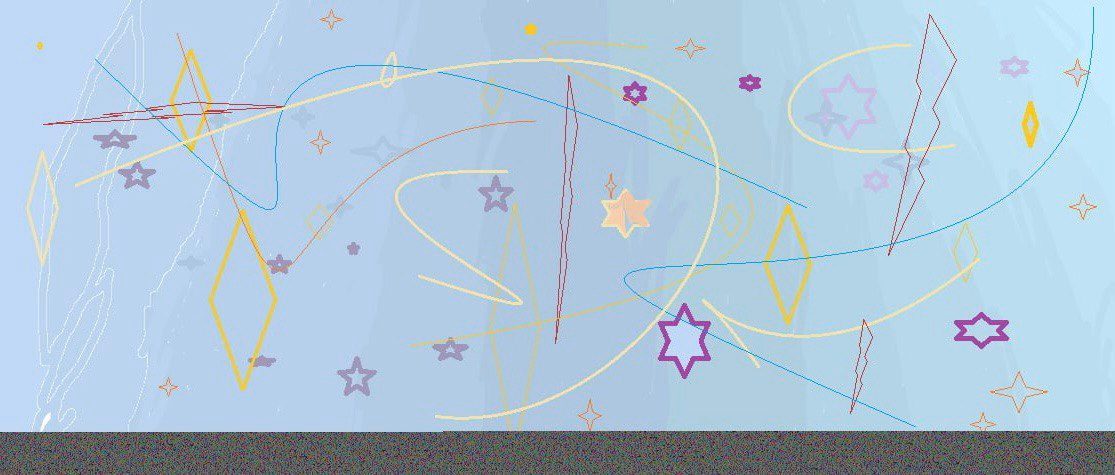
Однако при повороте, отзеркаливании (рис. 8) или любом другом механическом изменении закодированного изображения декодер работает неверно, так как механическая обработка изображения разрушает сообщение.

Рисунок 8 – отзеркаленное закодированное изображение.

# 6. Вывод

В работе был изучен метод стеганографии. Разработанная программа способна скрывать данные в изображенях, что позволяет передавать информацию через графические файлы. Была успешно скрыта информация из текстового и исполнимого файла, сохраняя при этом их функциональность. Также создана программа декодера, позволяющая извлечь исходное сообщение. Таким образом, получено первое представление об еще одной технологии, используемой для сокрытия информации, называемой стеганографией.

# 7. Контрольные вопросы

*1)Какие функции выполняет каждый элемент стегосистемы?*

**Стегоконтейнер** – информационная последовательность, в которой скрывается сообщение. Например, в его качестве выступает изображение, в котором и будет сокрыто сообщение. Различают потоковый (в кодере заранее не известно, достаточно ли объема контейнера для передачи всего сообщения) и фиксированный (размеры и характеристики контейнера известны заранее, что позволяет оптимально упаковывать данные) стегоконтейнер.

**Предварительный кодер** – устройство, предназначенное для преобразования скрываемого сообщения к виду, удобному для встраивания в стегоконтейнер.

Предварительная обработка частот выполняется с использование ключа K

для повышения секретности встраивания.

**Детектор** – устройство, предназначенное для определения наличия

стегосообщения.

**Декодер** – устройство, восстанавливающее скрытое сообщение. Нужно

понимать, что данный узел может отсутствовать.

*2)Перечислите основные виды атак на стегосистемы.*

* *Атаки на сообщение*. Направлены на извлечение стегосообщения без

искажения контейнера. Примеры атаки: очистка сигналов от шумов, нелинейная

фильтрация

* *Атаки на стегоконтейнер*. Направлены на удаление или порчу

сообщения путем различных преобразований, проводимых над стегоконтейнером.

Примеры атак: изменение контрастности, сжатие изображения, добавление шумов

* *Атаки на детектор*. Направлены на затруднение работы или

блокировка детектора. При этом сообщение в стегоконтейнере остается, но

теряется возможность его приема. Примеры атак: масштабирование, сдвиги, повороты, перестановка пикселей.

* *Атаки на механизм использования стегосообщений*. В основным связаны создание ложных стегосообщений, добавление нескольких сообщений.
* *Атаки на дополнительный ЦВЗ.* Противостояние попыткам защиты от

атак аффинного преобразования (масштабирование, сдвиг, поворот, усечение изображения).

*3)Что означает термин “стойкость стегосистемы”?*

Безопасность стегосистем оценивается их стойкостью, т.е. способностью

стегосистемы скрывать от квалифицированного нарушителя факт передачи

сообщений, способностью противостоять попыткам нарушителя разрушить,

исказить, удалить скрытно передаваемые сообщения, а также способностью

подтвердить или опровергнуть подлинность скрытно передаваемой информации.

Стегосистема является стойкой, если нарушитель, наблюдая информационный обмен между отправителем и получателем, не способен обнаружить, что под прикрытием контейнера передается то или иное сообщение.

*4)Какое влияние оказывает сжатие графических изображений на*

*алгоритмы встраивания стегосообщений?*

Практически любой способ обработки изображений может привести к

разрушению части или всего строенного сообщения. Сжатие графических

изображений – не исключение. Декодировать начальное стегосообщение будет невозможно полностью.

*5)Каковы методы противодействия стеганографическим атакам?*

Для защиты от атак аффинного преобразования (масштабирование, сдвиг,

поворот, усечение изображения) можно использовать дополнительные ЦВЗ. Эти ЦВЗ не несут в себе информации, но используются для регистрации выполняемых нарушителем преобразований. Другой альтернативой является вложение ЦВЗ в визуально значимые области изображения, которые не могут быть удалены из него без существенной его деградации.

Другим методом защиты от подобных атак является блочный детектор.

Модифицированное изображение разбивается на блоки размером 12×12 или 16×16 пикселей, и для каждого блока анализируются все возможные искажения, т.е. пиксели в блоке подвергаются поворотам, перестановкам и т.д. Для каждого изменения определяется коэффициент ЦВЗ. Преобразование, после которого коэффициент оказался наибольшим, считается реально выполненным нарушителем. Таким образом, появляется возможность обратить внесенные нарушителем искажения.

# 8. Приложение

Листинг программы «Кодер»:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

unsigned int fileSize(FILE\* file) {

fseek(file, 0, SEEK\_END);

int s = ftell(file);

fseek(file, 0, SEEK\_SET);

return(s);

}

int coder() {

int deg;

printf("Введите степень упаковки вашего сообщения: ");

scanf("%d", &deg);

FILE\* in = fopen("sky.bmp", "rb");

FILE\* out = fopen("answer.bmp", "wb+");

FILE\* in\_text = fopen("code.txt", "r");

int text\_size = fileSize(in\_text);

int pick\_size = fileSize(in);

int maxsize = (pick\_size \* (unsigned int)deg / 8 - 54);

if (text\_size > maxsize) printf("Размеры вашего текста не позволяют полностью поместить текст в стегоконтейнер!\n");

else {

char ch = 0;

for (int i = 0; i < 49; i++) {

fread(&ch, 1, 1, in);

fwrite(&ch, 1, 1, out);

}

fread(&ch, 1, 1, in);

fwrite(&deg, 1, 1, out);

int size;

for (int i = 3; i >= 0; i--) {

size = text\_size;

size = size >> (8 \* i);

fread(&ch, 1, 1, in);

fwrite(&size, 1, 1, out);

}

char bit, newbit, imgbit = 255;

unsigned char img\_mask = 255 << deg;

unsigned char txt\_mask = ~(255 << deg);

while (!feof(in\_text)) {

fread(&bit, 1, 1, in\_text);

if (!feof(in\_text)) {

for (int i = 8 / deg - 1; i >= 0; i--) {

newbit = bit;

newbit = newbit >> deg \* i;

newbit = newbit & txt\_mask;

fread(&imgbit, 1, 1, in);

imgbit = imgbit & img\_mask;

imgbit = imgbit | newbit;

if (!feof(in\_text))

fwrite(&imgbit, 1, 1, out);

}

}

}

while (!feof(in)) {

fread(&imgbit, 1, 1, in);

if (!feof(in))

fwrite(&imgbit, 1, 1, out);

}

}

fclose(in\_text);

fclose(in);

fclose(out);

return 0;

}

int main() {

setlocale(LC\_ALL, "Rus");

coder();

return 0;

}

Листинг программы «Декодер»:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

int decoder() {

FILE\* answer, \* final;

answer = fopen("answer.bmp", "rb");

final = fopen("final.txt", "wb");

fseek(answer, 49, SEEK\_SET);

int deg = fgetc(answer);

int size\_file = 0, imgbit = 0, f = 0;

for (int i = 3; i >= 0; i--) {

fread(&imgbit, 1, 1, answer);

f = (unsigned int)imgbit;

f = f << (8 \* i);

size\_file = size\_file | f;

}

char img\_mask = ~(255 << deg);

for (unsigned int j = 0; j < size\_file; j++) {

char ch = 0;

for (int i = 8 / deg - 1; i >= 0; i--) {

if (!feof(answer)) {

fread(&imgbit, 1, 1, answer);

imgbit = imgbit & img\_mask;

imgbit = imgbit << deg \* i;

ch = ch | imgbit;

}

}

fwrite(&ch, 1, 1, final);

}

fclose(final);

fclose(answer);

return 0;

}

int main() {

setlocale(LC\_ALL, "Rus");

decoder();

return 0;

}