**МАОУ «Инженерный лицей НГТУ»**

Направление: Защита информации

**Стеганография в цифровых изображениях.**

Автор:

**Вахрушев Богдан Игоревич**

Группа Л11-3 МАОУ «Инженерный лицей НГТУ»

Ленинского района

города Новосибирска

Научный руководитель:

Муль Павел Фридрихович

Новосибирск 2022

**Содержание**

[Введение 3](#_Toc120062110)

[История стеганографии 4](#_Toc120062111)

[Основы стеганографии в изображениях 5](#_Toc120062112)

[Практическое исполнение в программном обеспечении 6](#_Toc120062113)

[Заключение 8](#_Toc120062114)

[Источники 9](#_Toc120062115)

# Введение

Стеганография менее известный способ передачи информации, чем криптография, но не менее интересный и полезный в применении. В отличие от криптографии, которая скрывает содержимое сообщения, стеганография скрывает сам факт существования этого сообщения. Проект актуален, так как стеганография широко применима в области шифрования и защиты данных, используется для цифровых подписей и является основой систем защиты авторских прав.

**Цель:** Исследование и программирование алгоритмов компьютерной стеганографии в изображениях.

**Объект:** Стеганография

**Предмет:** Алгоритмы шифрования данных в изображениях

**Задачи:**

1. Провести исследование алгоритмов компьютерной стеганографии.
2. Запрограммировать алгоритм стеганографии для практического использования.
3. Определить плюсы и минусы подобного шифрования данных.

# История стеганографии

Первым задокументированным примером использования стеганографии были так называемые восковые дощечки. Они использовались для того чтобы писать на них с помощью остро заточенных палочек, однако можно было оставить некоторое сообщение на дне дощечки. На основу после этого заливали воск и поверх воска оставляли второе сообщение. Только тот, кто знал секрет такой дощечки мог догадаться что нужно читать на самом деле не верхнее сообщение, а соскрести слой воска и прочитать то что написано на самом низу дощечки.



Также, в 16-м веке был задокументирован способ писать сообщения внутри вареных яиц. Специальным раствором наносили сообщение на скорлупу еще не сваренного яйца, после этого яйцо варилось, сообщение снаружи скорлупы пропадало. Однако, когда его разбивали, внутри оказывалось то самое сообщение, которое благодаря специальному составу раствора пропиталась сквозь скорлупу на само яйцо.

Одним из известных благодаря поп-культуре методов стеганографии появился вместе с созданием газет в начале XIX века. Печаталась определенного содержания заметка в газете и человек, который знал, что нужно читать, к примеру, каждую пятую букву определенной заметки или, например, вторую букву из каждого слова то он мог прочитать секретное послание, адресованное именно ему.

Также в XX веке существовал популярный метод скрытия информации - невидимые чернила. Специальный состав чернил, которые пропадают после того как их нанести на бумагу и появляются, например, только после нагрева или какого-то химического воздействия на бумагу.

В 90х годах ХХ века появилось такое направление как компьютерная стеганография, основанное на особенностях компьютерной платформы: скрытие данных в неиспользуемых областях форматов файлов, подмена символов в названиях файлов, текстовая стеганография и т. д. В то же время появляется и цифровая стеганография, вводящая дополнительные данные в цифровые мультимедиа-объекты (изображения, видео, аудио, текстуры 3д-объектов), вызывая искажения, находящиеся ниже порога чувствительности человека.

# Основы стеганографии в изображениях

Стеганография – способ передачи и хранения информации с учетом сохранения в тайне самого факта этой передачи (хранения). В отличие от криптографии, которая скрывает содержимое сообщения, стеганография скрывает сам факт существования этого сообщения. Стеганографию часто объединяют с криптографией для повышения уровня безопасности сообщения.

Задача стеганографии в цифровых изображениях состоит в следующем: зашить в изображение сообщение так, чтобы это не было заметно человеческим глазом.

Выделяют несколько основных алгоритмов для выполнения данной задачи:

* + - 1. **LSB (англ. Last Significant Bit – Наименее значимый бит).** Наиболее распространенный алгоритм. Заключается в выделении наименее значимых бит изображения и заменой их на биты скрываемого сообщения.
      2. **PDV (англ. Pixel Value Difference – Разность значений пикселей).** Этот метод отличается от LSB тем, что учитывает перепады яркости изображения для более тщательного скрытия, а также берет блоки из нескольких пикселей, сглаживая видимые последствия шифрования.
      3. **GLM (англ. Grey Level Modification – Изменение уровня серого).** Метод заключается в изменении четности значения яркости изображения в черно-белом представлении. Каждый пиксель может сохранить 1 бит скрываемого сообщения.
      4. **MPV (англ. Mid Position Value – Значение в средней позиции).** Более сложен, чем остальные методы. Заключается во множестве преобразований изображения, основным из которых является преобразование Арнольда. Может содержать 2 бита зашифрованной информации в пикселе.

Каждый из алгоритмов выигрывает либо в объеме сохраняемой памяти, либо в чистоте изображения, но проигрывает в другом.

# Практическое исполнение в программном обеспечении

Для практической части своего проекта я выбрал алгоритм наименее значимого бита (LCB). Он позволяет записывать 1 байт информации в один пиксель, что равняется весу одного символа в кодировке ASCII.

Ссылки на код проекта находятся в приложении №3.

Алгоритм заключается в следующих шагах:

Для примера возьмем запись символа **‘C’ (01000011 в таблице ASCII)** в полностью белый пиксель  
**(R = 255 = 111111112, G = 255 = 111111112, B = 255 = 111111112).**

1. Разбивка байта символа на 3 части. В своем проекте я использовал разбивку 3|2|3. В нашем примере это будет **010|00|001**.
2. Очистка наименее значимых битов каждого цветового канала. В примере с белым пикселем это будет выглядеть так:

**R = 11111*000*2 G = 111111*00*2 B = 11111*000*2**

1. Вписывание в очищенные биты частей битов символа.  
   **R = 11111*010*2 G = 111111*00*2 B = 11111*011*2**

Эти значения байтов и будут получившимися значениями цветовых каналов пикселя с зашифрованным символом. Для расшифровки этого символа нужно выполнить действия в обратном порядке – выделить в байтах цветовых каналов крайние биты с учетом разбивки из 1 шага и собрать в исходный символ.

Также для равномерного распределения зашифрованных пикселей и дополнительной безопасности информации в своем проекте я использовал генератор псевдослучайных чисел, определяющий положение и порядок зашифрованных пикселей, «зерно» которого – ключ, задаваемый пользователем. Таким образом, даже зная, что в изображении находится скрытое сообщение и имея разработанное мною программное обеспечение, сторонний пользователь, не имея ключа шифрования, не сможет получить доступ к скрытой информации. В больших изображениях метод перебора ключей крайне неэффективен, что делает алгоритм защищенным от взлома.

Говоря про размеры файла, открывается минус данного алгоритма: он работает лишь с несжатыми форматами изображений, такими как BMP. Но, по моему мнению, это не является большой проблемой. Приведем пример: изображение 1000 х 1000 пикселей в формате BMP весит ~3Мб и может вместить в себя 1 000 000 символов скрытой информации. Для сравнения, в произведении «Война и мир» около 2 966 500 символов, что вмещается в 3 подобных картинки. При этом размер изображения не меняется, так как алгоритм лишь изменяет имеющуюся информацию, не добавляя новую. В изображении шифрованная информация создает легкий шум, не бросающийся в глаза (см. Приложение 1).

Также в ходе своей работы я реализовал алгоритм сокрытия в изображении другого изображения:

Он заключается в присвоении наименее значимым битам изображения наиболее значимые биты второго изображения

Для примера возьмем зашифровку пикселя цвета

**(R = 205 = 110011012, G = 92 = 010111002, B = 92 = 010111002).**

в полностью белый пиксель  
**(R = 255 = 111111112, G = 255 = 111111112, B = 255 = 111111112).**

1. Очистка двух наименее значимых битов каждого цветового канала. В примере с белым пикселем это будет выглядеть так:

**R = 111111*00*2 G = 111111*00*2 B = 111111*00*2**

1. Выделение наиболее значимых битов в шифруемом пикселе.  
   **R = *11*0011012 G = *01*0111002 B = *01*0111002**
2. Вписывание в очищенные биты основного пикселя биты шифруемого пикселя.  
   **R = 111111*11*2 G = 111111*01*2 B = 111111*01*2**

Эти значения байтов и будут получившимися значениями цветовых каналов пикселя с зашифрованным пикселем второй картинки. Для расшифровки этого изображения нужно выполнить действия в обратном порядке – выделить в байтах цветовых каналов два крайних бита и собрать их в исходный пиксель, поставив на место ведущих битов в пустом пикселе (в пикселе со значением нуля во всех каналах).

В итоге можно заметить, что исходное изображение практически не потеряло в качестве, а шифруемое изображение имеет достаточное качество для различения сути картинки (см. Приложение 2). При использовании такого метода шифрования для передачи QR-кодов или текста (изображенного графически) проблем не возникает вовсе, так как такой тип информации не нуждается в высоком качестве передачи.

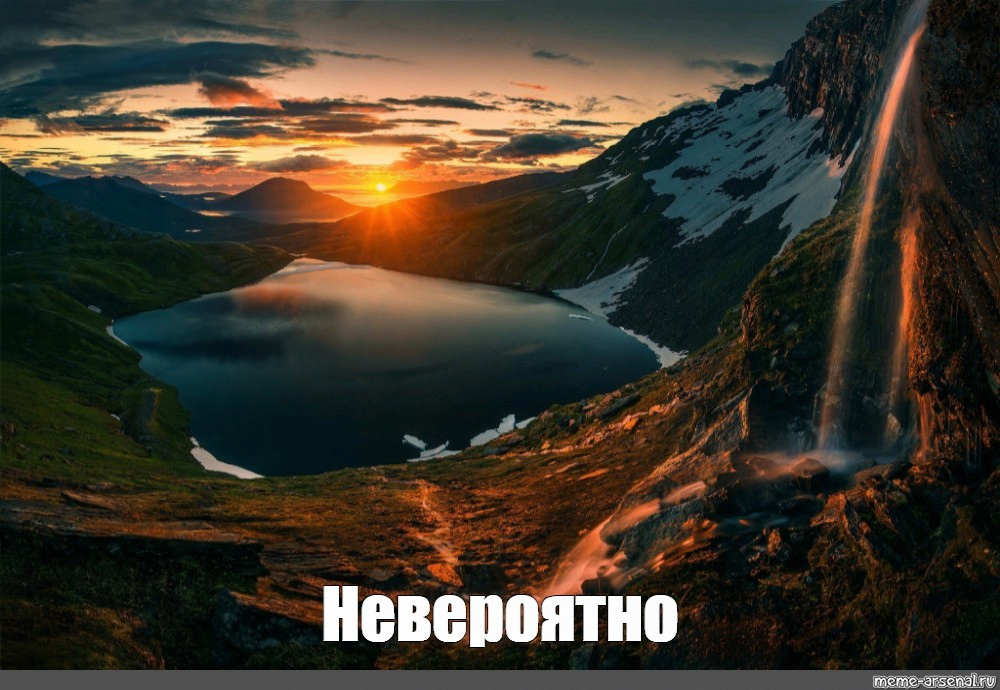
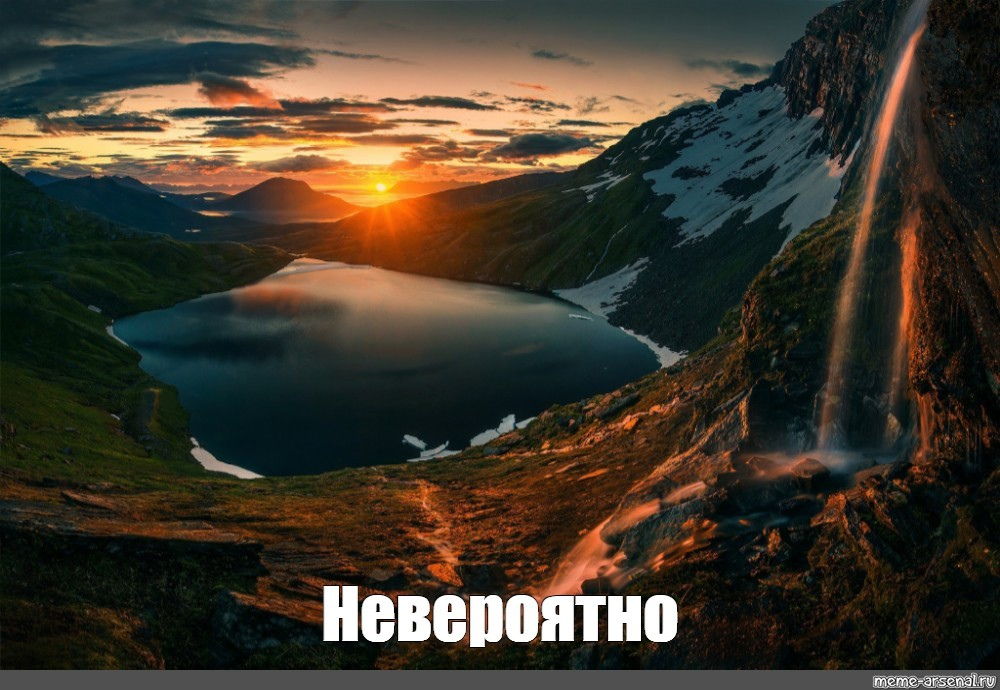
# Заключение

Стеганография в цифровых изображениях - хороший способ защищенной тайной передачи информации. В ходе работы были протестированы алгоритмы шифрования данных в изображениях. При использовании данных алгоритмов, с искажениями, находящимися ниже порога чувствительности человека, в изображение можно вместить довольно большой объем текстовой информации или даже отдельное изображение, размером меньше или равному исходному, с качеством, достаточным для различения сути шифруемой картинки.

Компьютерная стеганография – перспективное направление шифрования данных, которое можно использовать в системах защиты авторских прав, в тайной передаче информации.

# Источники

* История стеганографии - https://steganography-rsvpu.tilda.ws/
* Основная информация про стеганографию - https://ru.wikipedia.org/wiki/Стеганография\_в\_цифровых\_изображениях, https://ru.wikipedia.org/wiki/Стеганография
* Разборы принципов действия алгоритмов - https://habr.com/ru/post/114597/, https://habr.com/ru/post/140373/
* Разборы алгоритмов, теоретическая информация, история - https://photodb.illusdolphin.net/media/4781/stego.pdf

Приложение 1.1. Пустое изображение и с зашифрованным символом в каждом пикселе 

Приложение 1.2. Пустое изображение и с зашифрованным символом в каждом пикселе (приближенное)



Оригинал Шифрованное

Приложение 2.1. Изображение со спрятанным внутри изображением и расшифрованное изображение



Приложение 3. Ссылки на код и исходные данные проекта.



https://github.com/MrBreadnt/steganointerface