МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Исследование стеганографического метода на основе преобразования наименее значащих битов

Студент: Евсеенко В. П.

ФИТ 3 курс 4 группа

Преподаватель: Савельева М. Г.

Минск 2025

Содержание

[Введение 3](#_Toc201143709)

[1. Теоретические сведения 4](#_Toc201143710)

[2. Ход работы 5](#_Toc201143711)

# Введение

Цель лабораторной работы:

* изучение стеганографического метода встраивания/извлечения тайной информации с использованием электронного файла-контейнера на основе преобразования наименее значащих битов (НЗБ), приобретение практических навыков программной реализации данного метода.

Задачи:

* закрепить теоретические знания из области стеганографического преобразования информации, моделирования стеганосистем, классификации и сущности методов цифровой стеганографии;
* изучить алгоритм встраивания/извлечения тайной информации на основе метода НЗБ (LSB – Least Significant Bit), получить опыт практической реализации метода;
* разработать приложение для реализации алгоритма встраивания/извлечения тайной информации с использованием электронного файла-контейнера на основе метода НЗБ;
* познакомиться с методиками оценки стеганографической стойкости метода НЗБ.

# Теоретические сведения

Стеганографическая система (stegosystem, стегосистема или стеганосистема – в русскоязычной тематической литературе используются оба сокращения) – совокупность средств и методов, которые используются для формирования скрытого канала передачи (или хранения) информации.

Абстрактно стеганографическая система обычно определяется как некоторое множество отображений одного пространства (множества возможных сообщений *М*) в другое пространство (множество возможных стеганосообщений *S*), и наоборот.

Основные компоненты стеганосистемы:

• контейнер *С* (файл-контейнер или электронный документ произвольного формата), в котором размещается (осаждается, скрывается) тайное сообщение *М*; именно контейнер является упомянутым скрытым каналом;

• тайное сообщение *М*, осаждаемое в контейнер для передачи или хранения (например, с целью доказательства или защиты авторских прав на документ-контейнер; здесь речь может идти о невидимых цифровых водяных знаках (ЦВЗ));

• ключи, или ключевая информация, *K* системы, выполняющие ту же функцию, что и криптографические ключи; ключей может быть несколько, в соответствии с этим современные стеганосистемы характеризуют как многоключевые: один ключ отождествляется с методом встраивания/извлечения тайной информации, другой – с выбором элементов (например, битов) контейнера для его модификации при осаждении тайной информации, третий – для предварительного (перед встраиванием) преобразования тайной информации (например, на основе помехоустойчивого кодирования, сжатия или зашифрования) и т. д.;

• контейнер со встроенным сообщением, или стеганоконтейнер, *S*, который передается по открытому каналу, также являющемуся важным компонентом анализируемой системы; стеганоконтейнер будем именовать также стеганосообщением;

• для полноты упомянем также субъектов системы: отправителя и получателя.

В зависимости от формата документа-контейнера цифровую (или компьютерную) стеганографию подразделяют на классы:

• аудиостеганография;

• видеостеганография;

• графическая стеганография;

• текстовая стеганография;

• и др.

Стеганографической системой ∑ будем называть совокупность сообщений M*,* контейнеров *C*, ключей *K*, стеганосообщений (заполненных контейнеров) *S* и преобразований (прямого *F* и обратного *F*–1), которые их связывают:

∑ = (*M*, *C*, *K*, *S*, *F*, *F–*1).

Метод НЗБ основывается на ограниченных способностях зрения или слуха человека, вследствие чего людям тяжело различать незначительные вариации цвета или звука.

# Практическое задание

Было необходимо разработать приложение, реализующее метод НЗБ.

Функция embedMessage принимает на вход оригинальный текст, ссылку на оригинальный контейнер, ссылку на выходной контейнер со встроенным сообщением, а также метод встраивания. Код функции представлен в листинге 2.1.

|  |
| --- |
| export const embedMessage = async (*containerPath*: string, *message*: string, *outputImagePath*: string, *method*: Method) => {      const containerImage = await Jimp.read(*containerPath*);      const messageBytes = Buffer.from(*message*, 'utf8');      const messageBits = buf2bin(messageBytes);      const maxMessageBits = (containerImage.getWidth() \* containerImage.getHeight()) \* 3;      if (messageBits.length > maxMessageBits)          throw new Error("Message is too large for the container");      let messageBitIndex = 0;      let endZeroGroup = 0;      let xMax = 0;      let yMax = 0;      switch (*method*) {…}      for (let y = 0; y < yMax; y++) {          for (let x = 0; x < xMax; x++) {              let pixelColor: number = 0;              switch (*method*) {…}              const red: number = Jimp.intToRGBA(pixelColor).r;              const green: number = Jimp.intToRGBA(pixelColor).g;            const blue: number = Jimp.intToRGBA(pixelColor).b;              let newRed: number = 0;              let newGreen: number = 0;              let newBlue: number = 0;              if (messageBitIndex < messageBits.length) {                  if (+messageBits[messageBitIndex]) {                      newRed = red | 1;                  } else {                      newRed = red & ~1;                  }                  messageBitIndex++;                  if (messageBitIndex < messageBits.length) {…}                  if (messageBitIndex < messageBits.length) {…}                  switch (method) {                      case Method.ROWS:                          containerImage.setPixelColor(Jimp.rgbaToInt(newRed, newGreen, newBlue, 255), x, y);                          break;                      case Method.COLS:                          containerImage.setPixelColor(Jimp.rgbaToInt(newRed, newGreen, newBlue, 255), y, x);                          break;                  }              } else {                  if (endZeroGroup >= 3) {                      break;                  } else {…}          if (messageBitIndex >= messageBits.length) {              break;          }      }      await containerImage.writeAsync(outputImagePath);  }; |

Листинг 2.1 – Функция встраивания сообщения

Функция extractMessage принимает на вход ссылку на контейнер со встроенным сообщением, метод встраивания и извлекает его из контейнера. Код функции представлен в листинге 2.2.

|  |
| --- |
| export const extractMessage = async (*imagePath*: string, *method*: Method) => {      const containerImage = await Jimp.read(*imagePath*);      let messageBits: string = "";      const messageBitsLength = (containerImage.getWidth() \* containerImage.getHeight()) \* 3;      let messageBitIndex = 0;      let endZeroGroup = 0;      let xMax = 0;      let yMax = 0;      switch (*method*) {…}      for (let y = 0; y < yMax; y++) {          for (let x = 0; x < xMax; x++) {              let pixelColor: number = 0;              switch (*method*) {                  case Method.ROWS:                      pixelColor = containerImage.getPixelColor(x, y);                      break;                  case Method.COLS:                      pixelColor = containerImage.getPixelColor(y, x);                      break;              }              const red = Jimp.intToRGBA(pixelColor).r;              const green = Jimp.intToRGBA(pixelColor).g;              const blue = Jimp.intToRGBA(pixelColor).b;              messageBits += red & 1;              messageBitIndex++;              if (messageBitIndex < messageBitsLength) {                  messageBits += green & 1;                  messageBitIndex++;              }              if (messageBitIndex < messageBitsLength) {                  messageBits += blue & 1;                  messageBitIndex++;              }              if (messageBits.slice(messageBits.length - 3) === "000") {…}          if (messageBitIndex >= messageBitsLength) {              break;          }          if (endZeroGroup >= 3) {              break;          }      }      return convertBinaryToString(messageBits);} |

Листинг 2.2 – Функция извлечения сообщения

Результат работы приложения с исходным текстом «Evseenko Viktoriya Pavlovna» и внедрением по строкам представлен на рисунке 2.3.

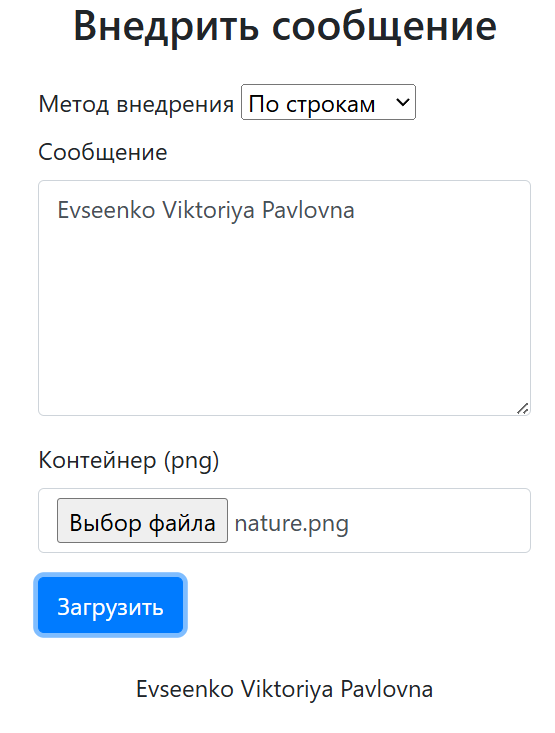


Рисунок 2.3 – Результат работы приложения

Исходный контейнер nature.png представлен на рисунке 2.4.



Рисунок 2.4 – Исходный контейнер

Контейнер со встроенным сообщением представлен на рисунке 2.5.



Рисунок 2.5 – Контейнер со встроенным сообщением

Визуально эти два контейнера ничем не отличаются. Для обнаружения следов осаждения сообщения необходимо сформировать цветовые матрицы.

Цветовая матрица исходного контейнера представлена на рисунке 2.6.



Рисунок 2.6 – Цветовая матрица исходного контейнера

Цветовая матрица контейнера с внедренным сообщением представлена на рисунке 2.7.



Рисунок 2.7 – Цветовая матрица контейнера с внедренным сообщением

В левом верхнем углу можно заметить различия. Различия представлены на рисунке 2.8. Сверху указан исходный контейнер, снизу – контейнер с внедренным сообщением.





Рисунок 2.8 – Следы осаждения сообщения на цветовой матрице при записи по строкам

Результат работы приложения исходным текстом которого является текстовая часть отчета по лабораторной работе №12 и внедрением по столбцам представлен на рисунке 2.9.

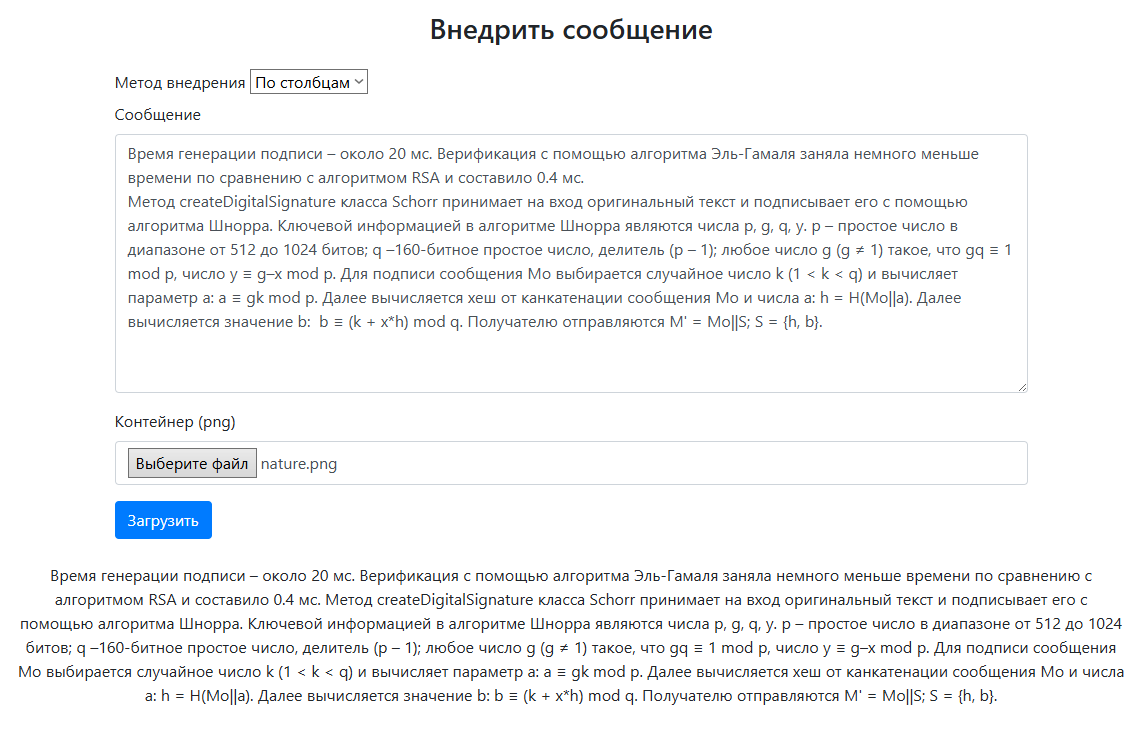


Рисунок 2.9 – Результат работы приложения

Различия в цветовых матрицах контейнеров при встраивании по столбцам представлены на рисунке 2.10. Слева указан исходный контейнер, справа – контейнер с внедренным сообщением.

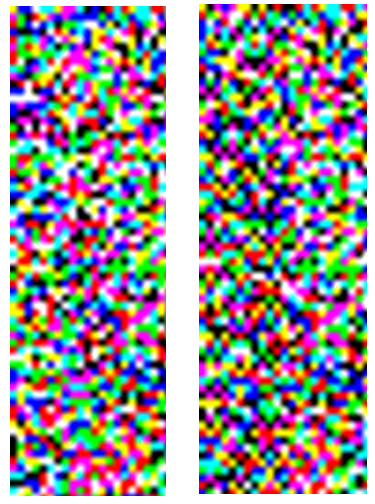


Рисунок 2.10 – Следы осаждения сообщения на цветовой матрице при записи по столбцам

Проведенный визуальный анализ показывает, что в большинстве случаев визуальных искажений в обработанных изображениях не наблюдается, что подтверждает эффективность стеганографического метода на основе преобразования наименее значащих битов. Также было выявлено, что метод внедрения сообщения по строкам или по столбцам не оказывает существенного влияния на визуальные характеристики конечного изображения.

# Вывод

В ходе выполнения лабораторной работы были изучены принципы цифровой стеганографии, особенности методов скрытия информации и классификация стеганосистем. Особое внимание было уделено методу наименее значащих битов (LSB), который позволяет незаметно внедрять тайные данные в электронные файлы-контейнеры, такие как изображения.

Был подробно рассмотрен и реализован алгоритм встраивания и извлечения скрытой информации с использованием метода НЗБ. Практическая реализация данного метода позволила закрепить полученные теоретические знания и получить навыки программирования стеганографических систем.

Также была проведена оценка стеганографической стойкости метода LSB, что позволило сделать вывод о его достоинствах (простота реализации, малозаметность внедрения) и ограничениях (низкая устойчивость к форматированию и сжатию файлов).

Таким образом, поставленные цели и задачи лабораторной работы были успешно достигнуты, а полученные знания и навыки могут быть использованы для дальнейшего изучения и разработки более совершенных методов стеганографии.