Ссылка на статью:
// Ученые записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2022, № 1, с. 34-42.

Поступила: 15.04.2022
Окончательный вариант: 05.05.2022

УДК 004.056.55

О методах защиты документов на основе JavaScript

Осипенко И.Н.

i.osipenko@ritg.ru
УлГУ, Ульяновск, Россия

В статье рассмотрены методы защиты документов на основе JavaScript. В процессе исследования подробно обсуждаются особенности использования стеганографии. Реализован алгоритм стеганографической защиты информации по методу LSB в среде Steganography.js, предложены способы повышения робастности и уровня сокрытия информации.

***Ключевые слова**: стеганография, изображение, алгоритм, JavaScript*

Введение

В современном информационном обществе, в условиях цифровизации и перехода к Индустрии 4.0 множество услуг обеспечивается с помощью компьютерных сетей и информационных технологий. При хранении, передаче или использовании информации экономического, технологического или государственного характера, возникает потребность в ее сокрытии от посторонних глаз, обеспечении сохранности и конфиденциальности. Чем больше ценность информации, тем большей защиты она требует.

Информация, представленная в цифровом виде, должна быть надежно защищена от многих угроз: несанкционированного доступа и использования, уничтожения, подделки, утечки, нарушения лицензионных соглашений, отказа от авторства и т.д. С целью достижения этой цели необходимо обеспечить защиту данных, используя разные алгоритмы шифрования.

Обозначенные обстоятельства, а также необходимость решения задач защиты прав собственности на информацию, представленную в цифровом виде, несомненно, актуализируют потребность проведения исследований в этом направлении. Однако, необходимо отметить, что на сегодняшний день не существует абсолютно надежного способа защитить

информацию. На данный момент разработки в этом направлении недостаточно систематизированы, а отечественные стандарты только разрабатываются. В качестве инструментов для развития этого направления широко используются методы теории вероятностей и математической статистики, теории быстрых ортогональных преобразований, теории аппроксимации, теории кодирования, теории сложности, теории погрешностей, цифровой обработки сигналов и изображений, а также криптографические и стеганографические методы.

Критически анализируя имеющиеся на сегодняшний день приемы и подходы, эксперты сходятся во мнении, что лучшим способом защитить информацию является сокрытие самого факта ее существования, что позволяет сделать *стеганография*. Основное отличие стеганографии от других методов защиты информации заключается в сокрытии факта существования секретного сообщения в другом, не привлекающем внимание объекте – контейнере, используя для этого особенности построения самого контейнера [1].

Для реализации стеганографии на практике наибольшую популярность приобрели объектно-ориентированные языки, такие, например, как Java с богатством возможностей и мощностью C++, поэтому рассмотрение особенностей их использования на сегодняшний день является важной научно-практической задачей, которая и предопределила выбор темы данной статьи.

Предложения, касающиеся формирования многослойного алгоритма, который способен обеспечить многоступенчатую защиту документов, изображений, включая защиту от копирования, электронную подпись и стеганографическую надпись, сформулированы в работах Kilic, E. Evrensevdi, B; Ж.А. Мусиралиева, З.Т. Суранчиевой, А.С. Махановой; И.В. Пономорева, Д.И. Строкина, М.В. Куркиной [2-4].

Однако несмотря на то, что исследуемая проблематика на сегодняшний день активно обговаривается, существует еще целый комплекс открытых вопросов, которые требуют детальной проработки и более глубокого изучения.

Таким образом, с учетом вышеизложенного, цель статьи заключается в рассмотрении возможности защиты документов с применением и реализацией метода стеганографии на языке JavaScript.

Анализ существующих алгоритмов

На сегодняшний день существуют разные методы для защиты документов путем сокрытия информации в цифровом виде с помощью стеганографии. Из их числа можно выделить методы, основанные на добавлении информации к имеющимся файлам (к примеру, мультимедийные), что приводит к искажениям, не улавливаемым человеком, поэтому это не вызывает заметных изменений в восприятии входных файлов. Встроенное, или как его еще называют, скрытое сообщение, помещается в контейнер. Контейнер представляет собой любые данные, предназначенные для сокрытия шифруемого текста.

Каждый метод стеганографии имеет как сильные, так и слабые стороны.

Рассмотрим критерии сравнения:

– уровень восприятия: стеганограмма не должна распознаваться органами чувств человека;
– вместимость: критерий определяет размер вставляемого сообщения;
– робастность: встраиваемое изображение не должно искажаться при передаче и обработке файла;
– скрытность: определяет успешность метода скрытия;
– вид области: критерий области встраивания, временная или частотная;
– независимость от формата.

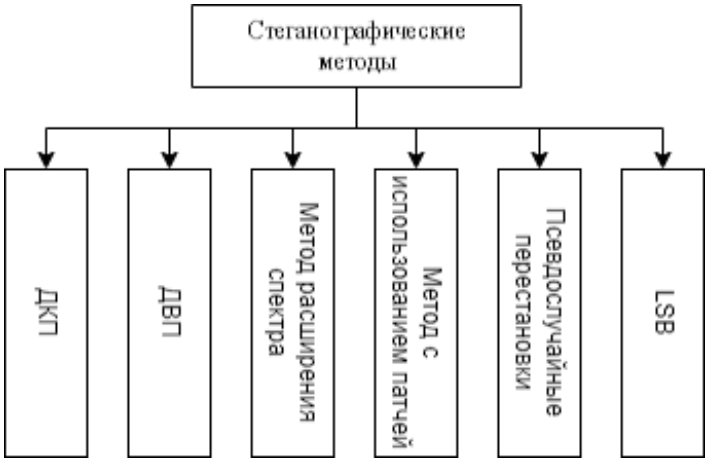

Рис. 1. Классификация стеганографических методов

В идеальных условиях алгоритм должен удовлетворять высоким уровням всех критериев.

Рассмотрим пригодность различных алгоритмов для форматов файлов:

LSB для BMP. Растровый формат, не использует сжатия. Для сокрытия сообщения в этих файлах необходим очень большой контейнер.

LSB для JPEG. Распространенный формат, использует 8 битов на каждый цвет RGB. Может скрыть сообщение большого объема.

LSB для GIF. Формат кодирует пиксель 8 битами, изображение записывается в 256 цветах. Алгоритм LSB скрывает информацию с различными степенями успеха в зависимости от доли изменяемых бит.

Псевдослучайные перестановки. Метод вставляет биты сообщения с изменением порядка их появления в сообщении.

Метод с использованием патчей. Недостаток этого метода состоит в том, что в один патч инкапсулируется только один бит [5]. Преимущество метода состоит в том, что сообщение распределено по всему изображению.

Метод расширения спектра. Этот метод распределяет сообщение по всему изображению. Частотная характеристика сообщения обладает гораздо меньшей энергией, чем энергия контейнера.

Дискретное косинус-преобразование (ДКП). Методы области преобразования (частотной области) скрывают сообщение в значительной области изображения, что делает их более робастными по сравнению с методами во временной области, включая сжатие, обрезку и некоторые алгоритмы обработки изображений.

Дискретное вейвлет-преобразование (ДВП). Инкапсуляция сообщения с помощью ДВП дает хорошие результаты, которые превосходят методы ДКП.

Основные форматы графических файлов имеют различные методы сокрытия с сильными и слабыми сторонами.

Алгоритм LSB

Для разработки алгоритма защиты информации на основе языка программирования JavaScript с использованием стеганографии представляется целесообразным применить метод наименьшего значимого бита (LSB) по следующим причинам:

- Мультимедиаконтейнеры не вызывают подозрений, можно без проблем отправить свою фотографию получателю;

- Младшие биты оцифрованных изображений могут иметь различное распределение в зависимости от применявшихся параметров аналого-цифрового преобразования, дополнительной компьютерной обработки и т.д. Данная особенность делает метод наименьших значащих битов наиболее защищенным от обнаружения встраивания;

- Реализация LSB для большинства стандартов файлов-контейнеров не требует значительных затрат времени и сил;

- Алгоритм легко подвергается повышению надежности сокрытия;

- Высокий уровень вместимости.

Суть метода LSB заключается в замене последних значимых битов в контейнере на биты сообщения, которое скрывается. Разница между заполненным и пустым контейнерами должна быть незаметной [6].

На рис. 2 показан первый этап процесса, когда доступ к данным изображения осуществляется в виде серии байтов. В зависимости от формата изображения, пиксель может быть представлен одним или несколькими байтами.

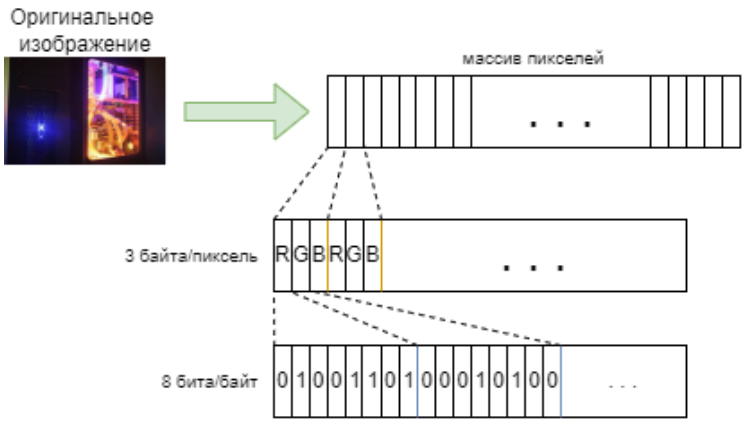

Рис. 2. Доступ к битам изображения PNG

Следующий этап — чтение текстового файла и доступ к его битам (рис. 3).

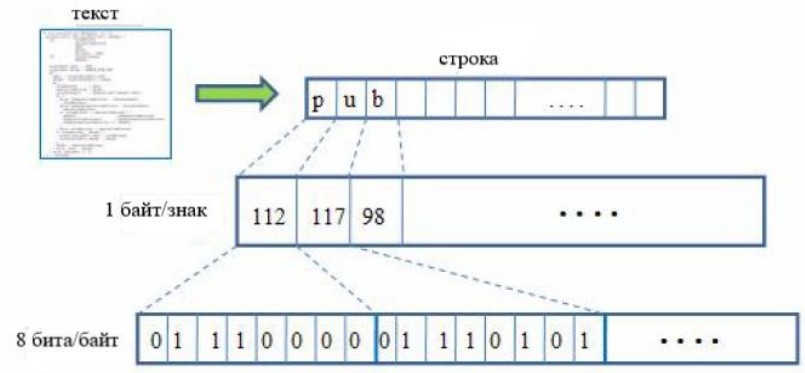

Рис. 3. Доступ к битам текстового файла

Итак, реализация алгоритма по LSB-методу включает в себя следующие этапы:

1) начало работы программы, получение входных данных – файла-контейнера;

2) получая на входе файл с данными, они определенным образом обрабатываются и превращаются в последовательность битов для дальнейшего встраивания в графический файл;

3) разбиение полученных байтов изображения на новые пары, по несколько битов по каждому каналу; в результате получается новый оттенок, очень похожий на предыдущий; эту разницу трудно заметить даже при большой площади заливки и согласно результатам исследований, замена двух младших битов не воспринимается человеческим глазом. Аналогично можно занять три разряда, но это немного исказит качество картинки. Данный процесс выполняется в цикле, чтобы равномерно распределить нужные биты по файлу, и чтобы встраивание не бросалось в глаза, что значительно повышает безопасность сохраненной информации и относительно сохраняет качество входного контейнера-картинки.

4) обработка ошибок при записи/чтении файла; при условии, что при проверке получено положительное значение – найден сбой, мусор и т.п., – необходимо предпринять ряд шагов для устранения обнаруженных отклонений, после чего повторно происходит прогон по условию и возвращение к главной части процесса кодирования данных;

5) при завершении работы и встраивании всех необходимых данных в изображение следует корректно переписать файл-контейнер, а также предусмотреть дополнительный файл для декодирования.

Для повышения уровня сокрытия информации и повышения робастности алгоритмов при реализации рассмотрены следующие способы:

- необходимо внедрять сообщение в случайные байты;
- необходимо максимально уменьшать объем внедряемой информации;
- необходимо выбирать изображения с высоким уровнем шума;
- необходимо использовать нигде не появлявшиеся изображения;

– необходимо периодически использовать ±1 кодирование: уменьшение или увеличение всего байта на единицу в случайном порядке, вместо изменения значащего бита в байте цвета (за исключением максимального и минимального значений байта);
– необходимо предварительно оценивать объем встраиваемого сообщения и размер исходного файла.

Следующим этапом является выбор среды разработки и языка программирования. Для выполнения поставленной задачи используем одно из существующих решений стеганографии, а именно Steganography.js, которое содержит базовый набор методов для скрытия текста внутри изображения и последующего извлечения текста.

Steganography.js — это библиотека JavaScript, используемая для кодирования секретных сообщений внутри изображений и их повторного декодирования. Steganography.js использует алгоритм для преобразования заданного сообщения в соответствующие двоичные данные, которые затем будут скрыты в альфа-канале выбранного изображения. Затем для обработки данных и изображения используется элемент HTML5 canvas.

Класс Steganography реализует метод LSB для скрытия текста внутри PNG-изображения, но с одним дополнением. Длина текста в двоичном виде вычисляется заранее и скрывается в изображении перед текстом. Другими словами, стеганографическая информация (стего) состоит из двух частей: размер двоичного сообщения, а затем само сообщение. Стего распространяется по изображению путем модификации LSB каждого байта. Это означает, что 1 байт данных стего требует модификации 8 байт изображения (т.е. 1 бит данных стего хранится в 1 байте изображения).

Информация о размере используется при извлечении текста из изображения, чтобы было понятно, когда следует остановиться. Данные о размере — это целое число, занимающее четыре байта, поэтому в начале изображения требуется 32 (4*8) байта.

Класс Steganography имеет два публичных статических метода:

– hide (textFnm, imFnm);
– reveal (imFnm).

Так как использованные методы являются публичными и подразумеваются общеизвестными, то секретный текст, встраиваемый в контейнер, должен быть предварительно зашифрован одним из криптографических алгоритмов, например RSA, Диффи-Хеллмана. Ключ этого алгоритма является секретным.

Пример работы алгоритма

Встраиваемый текст:

“Что это? я падаю! у меня ноги подкашиваются, — подумал он и упал на спину. Он раскрыл глаза, надеясь увидать, чем кончилась борьба французов с артиллеристами, и желая знать, убит или нет рыжий артиллерист, взяты или спасены пушки. Но он ничего не видал. Над ним не было ничего уже, кроме неба, — высокого неба, не ясного, но все-таки

неизмеримо высокого, с тихо ползущими по нем серыми облаками. “Как тихо, спокойно и торжественно, совсем не так, как я бежал, — подумал князь Андрей, — не так, как мы бежали, кричали и дрались; совсем не так, как с озлобленными и испуганными лицами тащили друг у друга банник француз и артиллерист, — совсем не так ползут облака по этому высокому бесконечному небу. Как же я не видал прежде этого высокого неба? И как я счастлив, что узнал его наконец. Да! все пустое, все обман, кроме этого бесконечного неба. Ничего, ничего нет, кроме его. Но и того даже нет, ничего нет, кроме тишины, успокоения. И слава Богу!..”

Рис. 3. Вид изображения до встраивания (523 КБ)

Рис. 4. Вид изображения после встраивания (527 КБ)

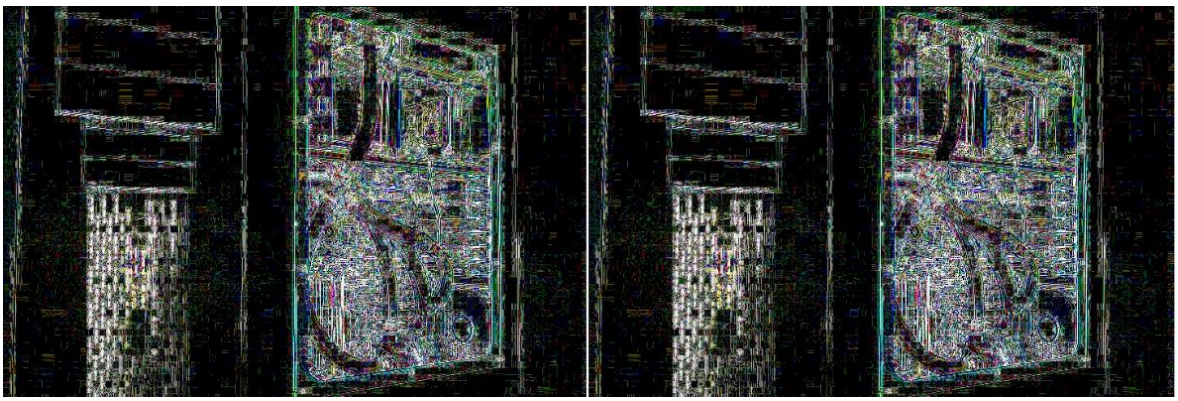

Рис. 5. Сравнение шума в изображении до и после встраивания

Заключение

Основным результатом представленной работы является реализация алгоритма стеганографической защиты информации по методу LSB в среде Steganography.js, при этом были предложены способы повышения робастности и уровня сокрытия информации. Алгоритм не изменяет визуальное качество изображения, что делает невозможным выявление факта сокрытия информации.

Список литературы

1. Khan M. A Novel Image Steganographic Approach for Hiding Text in Color Images using HSI Color Model // *Middle-East J. Sci. Res.* 2014, v. 22, no. 5, p. 647-654.
2. Kilic, E., Evrensevdi, B. A Review on the Different Types of Steganography // *Istanbul Technical University*. 2020, p. 1-8.
3. Мусиралиев Ж.А., Суранчиева З.Т., Маханова А.С. Программирование метода защиты информации // *Вестник Казахского национального женского педагогического университета*. 2018, № 1 (73), с. 47-52.
4. Пономарев И. В., Строкин Д. И. Стеганографические методы встраивания и обнаружения сокрытых сообщений, использующие GIF-изображения в качестве файлов-контейнеров // *Известия Алтайского государственного университета*. 2022, № 1(123), с. 112-115.
5. Morkel T., Eloff J. H.P., and Olivier M. S. An overview of image steganography. ICSA. 2005, p. 1-10.
6. Elshoura S. M. A secure high capacity full-gray-scale-level multi-image information hiding and secret image authentication scheme via Tchebichef moments // *Signal processing. Image communication*. 2013, no 5, p. 531-552.

On the metods to protect documents based on JavaScript

Osipenko, I.N.

i.osipenko@ritg.ru

Ulyanovsk State University, Ulyanovsk, Russia

The paper considers the methods to protect documents based on JavaScript. In the course of the study, the features of the use of steganography are considered in detail. The algorithm of steganographic protection of information by the LSB method in the Steganography environment is implemented.js, proposed ways to increase robustness and the level of information concealment.

Keywords*: steganography, image, algorithm, JavaScript*