МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
МЕХАНИКО – МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ
Кафедра дифференциальных уравнений и системного анализа

РУЦКАЯ
Елизавета Романовна

МЕТОДЫ ВНЕДРЕНИЯ ВОДЯНЫХ ЗНАКОВ В ЭЛЕКТРОННЫЕ ДОКУМЕНТЫ

Дипломная работа

Научный руководитель:
доктор технических наук,
профессор Липницкий В. А.

Допущена к защите
«___» _____2018 г.
Зав. кафедрой дифференциальных уравнений и системного анализа
доктор физ.-мат. наук, профессор В. И. Громак

Минск, 2018

ОГЛАВЛЕНИЕ

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

КС – компьютерная стеганография

ЦС – цифровая стеганография

ЦВЗ – цифровой водяной знак

ЦО – цифровой отпечаток

СВЗ – стеганографический водяной знак

СЭДО – система электронного документооборота

RGB – Red, Green, Blue

HTML – HyperText Markup Language

DLP – Data Leak Prevention

ПХ – преобразование Хаара

БПХ – быстрое преобразование Хаара

ШПС – шумоподобный сигнал

РЕФЕРАТ

В дипломной работе 48 страниц, 20 рисунков, 1 таблица, 12 источников, 4 приложения.

ЦИФРОВОЙ ВОДЯНОЙ ЗНАК, СТЕГАНОГРАФИЯ, ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЕ, КОНТЕЙНЕР, СТЕГОСООБЩЕНИЕ.

Целью дипломной работы является изучение и реализация основных методов внедрения цифровых водяных знаков в связи с недостаточным изложением таковых в открытых источниках информации.

В дипломной работе получены следующие результаты:

1) Реализован метод вставки младшего бита для контейнеров большого и малого размеров,
2) Реализован метод сокрытия бинарных данных в HTML-документе,
3) Реализовано внедрение водяного знака в PDF-документ.

Для работы была использована вычислительная среда Wolfram Mathematica 11.0, локальный сервер Denwer, среда программирования Intellij Idea, языки программирования Java SE и PHP.

Дипломная работа носит практический характер. Её результаты могут быть использованы для внедрения и извлечения цифровых водяных знаков и скрытых сообщений в изображениях и цифровых документах.

Дипломная работа выполнена автором самостоятельно.

РЭФЕРАТ

У дыпломнай рабоце 48 старонак, 20 малюнкаў, 1 табліца, 12 крыніц, 4 прыкладання.

ЛІЧБАВЫ ВАДЗЯНЫ ЗНАК, СЦЕГАНАГРАФІЯ, ВЭЙВЛЕТ-ПЕРАЎТВАРЭННЕ, КАНТЭЙНЕР, СЦЕГАПАВЕДАМЛЕННЕ.

Мэтай дыпломнай работы з’яўляецца вывучэнне і рэалізацыя асноўных метадаў укарнення лічбавых вадзяных знакаў у сувязі з недастатковым выкладаннем такіх у адкрытых крыніцах інфармацыі.

У дыпломнай рабоце былі атрыманы наступныя вынікі:

1) Рэалізаваны метад ўстаўкі малодшага біта для кантэйнераў вялікага і малога памераў,
2) Рэалізаваны метад утойвання бінарных дадзеных у HTML-дакуменце,
3) Рэалізавана ўкарненне вадзянога знака ў PDF-дакумент.

Для работы было выкарыстана вылічальнае асяроддзе Wolfram Mathematica 11.0, лакальны сервер Denwer, асяроддзе праграмавання Intellij Idea, мовы праграмавання Java SE і PHP.

Дыпломная работа мае практычны характар. Яе вынікі могуць быць выкарыстаны для ўкарнення і здабычы лічбавых вадзяных знакаў і схаваных паведамленняў у малюнках і лічбавых дакументах.

Дыпломная работа выканана аўтарам самастойна.

ABSTRACT

Thesis project is represented in the form of an explanatory note of 48 pages, 20 figures, 1 table, 12 references, 4 applications.

DIGITAL WATERMARK, STEGANOGRAPHY, WAVELET-TRANSFORMATION, CONTAINER, STEGOMESSAGE.

The purpose of the thesis project is to study and implement the basic methods of introducing digital watermarks in connection with the lack of information in open sources of information.

The main results of the thesis project are as follows:

1) The method of inserting the least significant bit for large and small containers implemented,
2) The method of hiding binary data in an HTML-document implemented,
3) The insertion of the digital watermark in a PDF-document implemented.

Wolfram Mathematica 11.0, Denwer local server, programming environment Intellij Idea, Java SE and PHP programming languages were used for the project.

This thesis project is of practical nature. Its results can be used to introduce and extract digital watermarks and hidden messages in images and digital documents.

Solely the author did the thesis project.

ВВЕДЕНИЕ

С возникновением вычислительной техники вопрос защиты информации становится всё более актуальным. Одним из её аспектов является защита авторских прав на электронные документы и ресурсы. Используя вычислительные средства, злоумышленники могут осуществить пользование и распространение электронной информации. Таким образом, правообладателю наносится и материальный, и моральный ущерб.

Самыми эффективными средствами защиты авторства электронных документов являются использование цифровой стеганографии, а также цифровых водяных знаков.

Целью данной дипломной работы является изучение и реализация основных методов внедрения цифровых водяных знаков в связи с недостаточным изложением таковых в открытых источниках информации.

Для осуществления поставленной цели были выполнены ниже перечисленные задачи:

1) Исследован и реализован метод вставки младшего бита для контейнеров различных размеров,
2) Исследовано и реализовано сокрытие бинарных данных в HTML-документе,
3) Исследовано многоуровневое вейвлет-преобразование. Реализовано внедрение водяного знака в PDF-документ.

Для работы была использована вычислительная среда Wolfram Mathematica 11.0, локальный сервер Denwer, среда программирования Intellij Idea, языки программирования Java SE и PHP. При реализации вышеупомянутых методов были использованы JPG изображения различных размеров, текстовые контейнеры длины до 1000 символов, а также разметка созданной автором дипломной работы HTML-страницы.

В главе 1 подробно раскрывается понятие стеганографии, ключевые понятия и методы стеганографии как науки, а также практическое применение стеганографии. В главе 2 подробно описывается суть методов вставки младшего бита при большом и малом размерах контейнера, внедрения бинарных данных в HTML-документ; описывается суть преобразования Хаара, двумерного и, как его следствие, многомерного вейвлет-преобразования. В главе 3 приведено подробное описание реализации автором дипломной работы указанных в задачах методов внедрения цифрового водяного знака.

ГЛАВА 1
СТЕГАНОГРАФИЯ

В ряде методов для защиты данных с помощью цифровых водяных знаков встроенная информация может, а порой должна быть визуально различимой. Поэтому термин «стеганография», или «встраивание информации», в употреблении является более предпочтительным, нежели термин «сокрытие информации».

Под термином «стеганография» понимается такая область знаний, которая охватывает широкий круг проблем с внедрением информации (секретного сообщения или цифрового водяного знака) в содержимое другого информационного объекта (открыто передаваемой информации или контейнера).

Как наука, стеганография является актуальной областью для исследований с точки зрения обеспечения безопасности информации, а также с точки зрения защиты персональных данных и авторских прав. Сейчас проводятся многочисленные исследования в области цифровой и компьютерной стеганографии. Они весьма разнообразны и направлены на улучшение характеристик стеговложений и возможность создания новых методов с использованием передовых достижений в различных отраслях знаний, открывающих большие перспективы для потребителей услуг связи. Несмотря на разнообразие вариантов, которые предлагают авторы в своих работах, основной задачей стеганографии по-прежнему остаётся поиск компромисса между скрытностью передачи и объёмом передаваемой информации. Направленные на поиск решений идеи, которые позволяют сформировать системы и сети стеганографической связи, отвечающие всем предъявляемым к ним требованиям, наравне с телекоммуникационными сетями, являются перспективными и целесообразными.

1.1. Ключевые понятия

К основным положениям для встраивания информации относятся следующие:

1) методы скрытия должны обеспечить аутентичность и целостность файла,
2) предположительно, атакующему хорошо известны возможные стеганографические методы,
3) безопасность методов должна базироваться на сохранении основных свойств передаваемого стеганографическим преобразованием открытого файла,
4) извлечение секретного сообщения должно представлять для противника сложную вычислительную задачу даже в случае, когда он знает о самом факте сокрытия такового.

Стеганография, как самостоятельное направление, выделилась недавно, поэтому полностью сформировавшейся терминологии не имеет. Главные понятия стеганографии были сформулированы на 1-ой Международной конференции по сокрытию данных в 1996 г. При этом, как ни парадоксально, само слово «стеганография» не имеет чёткого определения.

Тем не менее авторы статьи [4] привели термины, чьи определения сформулированы в ходе обобщения и уточнения понятий, наиболее часто употребляемых публикациями данной области:

- скрытая (стенографическая) передача информации – реализующие методы передачи информации процессы, при которых возможна и передача, и хранение дополнительной информации в структуре цифровых данных, которые используются в качестве контейнера за счёт своей избыточности,
- контейнер – некие цифровые данные, использование избыточности которых позволяет передавать дополнительную информацию, не обнаруживая самого факта передачи,

- пустой контейнер – контейнер, не имеющий никакой дополнительной информации, кроме, возможно, инициализационных данных,
- стегосистема – совокупность средств, а также методов передачи и приёма пустого контейнера, функционирующих взаимосвязано со средствами, используемыми для создания скрытого канала передачи информации.

Также автор статьи [3] дополнил существующие определения следующими:

- заполненный контейнер – контейнер, имеющий некоторую встроенную информацию,
- скрытое сообщение (стегосообщение) – сообщение, встраиваемое в контейнер,
- стегоканал – канал передачи скрытого сообщения,
- стегоключ – секретный ключ, который нужен для сокрытия информации и, соответственно, получения её.

Также в статье [3, с. 5, рисунок 2] изображён процесс передачи скрытого сообщения.

Выполнять функцию контейнера могут:

1) текстовые файлы,
2) графические файлы,
3) аудио файлы,
4) электронные документы и др.

Они содержат информацию, которую можно несколько изменить так, чтобы это изменение оказалось ещё недостаточным для того, чтобы быть воспринятым человеческими органами чувств, но уже достаточным для того, чтобы быть воспринятым программой-обработчиком. На этом и основывается их выбор.

Автор пособия [1] обобщил все эти знания и привёл 4 основные категории встраивания информации [1, с. 13, таблица 1.1].

1.2. Цифровая стеганография

Термин «цифровая стеганография» – это наука о надёжном скрытии одних битовых последовательностей в иных битовых последовательностях, имеющих аналоговую природу. Здесь незаметность подразумевает включение человека в стегосистему, надёжность – устойчивость к искажениям различных видов, а аналоговая природа – встраивание в оцифрованный непрерывный сигнал. Человек тут рассматривается как дополнительный приёмник информации, предъявляющий к стегосистеме довольно трудно формализуемые требования.

В русскоязычных работах, посвящённых стеганографической науке, часто используется термин цифровой водяной знак (ЦВЗ). Под ним подразумевают или стеганографический водяной знак (СВЗ), или цифровой отпечаток (ЦО) (а порой и СВЗ, и ЦО одновременно). Возникающие проблемы и задачи при реализации ЦО и СВЗ принципиально различны. Действительно, СВЗ во всех копиях электронного документа одинаковый, в то время как ЦО во всех копиях документа различается.

1.2.1 Методы цифровой стеганографии

В качестве науки цифровая стеганография оформилась в последние годы. В ней содержатся следующие направления:

- встраивание информации с целью её передачи,
- встраивание цифровых водяных знаков,
- встраивание идентификационных номеров,
- встраивание заголовков.

В пособии [2, с. 19, рисунок 4] автором приведена полная классификация методов цифровой стеганографии по принципу скрытия, по свойствам форматов файлов и т.д.

Отдельной классификационной группой среди стеганометодов является внедрение ЦВЗ и организация стегоканалов с использованием шумоподобных сигналов (ШПС), которые также именуются методами с расширением спектра. В сравнении со встраиванием в частотную и пространственную области, в этом случае модифицируется всё изображение целиком. Ведь изменению подлежат только элементы и коэффициенты, где зрение человека менее чувствительно, как и изменение статистики изображения.

1.2.2. Цифровые отпечатки

Данная разновидность стеганографии предполагает наличие стеганографических меток-сообщений в каждой копии контейнера. Например, ЦО таким образом могут применяться для защиты исключительного права. Если противник с помощью какого-нибудь алгоритма сможет извлечь ЦО из контейнера, то определить его будет невозможно. Пока противник не станет подделывать ЦО, то он не сможет без факта обнаружения распространять защищаемый контейнер. Таким образом, при извлечении ЦО третья сторона может преследовать две цели:

1) извлечение ЦО из контейнера (слабая цель),
2) подмена одного ЦО другим ЦО (сильная цель).

1.2.3. Стеганографические водяные знаки

В сравнении с ЦО, СВЗ предполагает наличие одинаковых меток в каждой копии контейнера. СВЗ также можно использовать для подтверждения авторских прав.

1.2.4 Цифровой водяной знак

Представление и передача цифровых сообщений дают преимущества, сводимые к нулю лёгкостью, с которой возможны их воровство и модификация. Встраивание в защищаемый объект особых невидимых меток является одним из эффективнейших средств для защиты электронной информации. Так как разработка над методами ЦВЗ началась абсолютно недавно, то здесь присутствует много неясных проблем требующих детального разрешения. ЦВЗ могут содержать какую-либо информацию собственника или аутентичный код либо управляющую информацию. Для защиты с ЦВЗ самыми подходящими объектами являются файлы аудио- и видеоданных, а также неподвижные изображения.

Этот метод получил название от хорошо известного способа защиты ценных бумаг, в том числе и денег от подделывания. ЦВЗ, отлично от обычных водяных знаков, могут быть и видимыми, и невидимыми.

ЦВЗ является представителем наиболее популярных направлений, получивших развитие в последнее время. Во многом это определяется необходимостью обеспечить защиту от несанкционированного распространения информации, которая является интеллектуальной собственностью. Организация стеганографических каналов является актуальным направлением для различных организаций и ведомств, где обеспечение безопасности является первостепенным требованием. Преследуемые при организации стегоканалов и при встраивании ЦВЗ цели различны. В связи с этим основные требования, которые устанавливаются перед разрабатываемыми стегометодами, имеют ряд отличий.

Так как ЦВЗ используется для защиты от несанкционированного доступа, то не является критичным знание нарушителя о факте встраивания ЦВЗ в защищаемый объект, что нельзя сказать о робастности ЦВЗ, поскольку основной атакой, применяемой в этой ситуации является геометрическая. Задача же

повышения скрытной пропускной способности при всём этом не ставится, в отличие от надобности обеспечения высокой достоверности приёма бит ЦВЗ.

Скрытность встраивания при организации скрытого канала является определяющим требованием, следовательно, основными атаками являются визуальная атака и статистический анализ. Поскольку при создании каналов речь идёт о передаче стегосообщения, необходимо обеспечить достоверность приёма скрытых данных и скрытую пропускную способность не хуже, чем минимальная зависимость от вида передаваемых сообщений.

1.3. Компьютерная стеганография

Под названием «компьютерная стеганография» понимается наука, основанная на использовании в качестве контейнера элементов и особенностей современного ПО, вычислительной техники.

В основном методы компьютерной стеганографии основываются на двух принципах:

1) существуют файлы, которым не нужна полная точность (изображения), и их можно подвергать изменениям без видимых потерь информативности,
2) органы чувств человека не способны различать незначительные отклонения в изменённых вышеуказанным методом файлах, а также нет специализированного оборудования для решения данной задачи.

Среди основных подходов реализации методов КС в рамках какой-либо информационной среды находится выделение малозначимых фрагментов данной среды и замена существующей во фрагментах информации данными, которые необходимо скрыть.

А так как в КС рассматриваются среды, которые поддерживаются вычислительными средствами и компьютерными сетями, то и вся информационная среда может в конечном итоге быть представлена в цифровом виде.

Фрагменты, малозначимые для «кадра информационной среды», заменяются фрагментами скрываемой информации относительно некоторого алгоритма или метода. «Кадр информационной среды» – это определённая часть среды, выделенная по характерным признакам. Такими признаками зачастую являются семантические характеристики выделяемой части информационной среды, такие как например, отдельное изображение, Web – страница или аудио файл.

1.3.1 Методы компьютерной стеганографии

По способу сокрытия методы компьютерной стеганографии разделяются на две категории: непосредственной замены, а также спектральные методы. Первые используют избыточность информационной среды во временной (для звука) или в пространственной (для изображения) области. Их суть лежит в замене незначительной части контейнера битами стегосообщения. Вторые же для сокрытия данных используют спектральные методы для представления элементов среды, куда встраиваются скрываемые данные.

Для компьютерной стеганографии основным направлением является именно использование свойств избыточности контейнера-оригинала. При этом следует учитывать тот факт, что в результате сокрытия информации происходит нарушение структуры контейнера или же происходит искажение некоторых его статистических свойств и целостности. Следовательно, необходимо уменьшать эффект демаскирующих признаков.

Отдельно можно выделить группу методов, которые используют особые свойства форматов представления файлов:

- зарезервированные методы расширения полей файлов, где поля наполняются нулями и не учитываются в программе,
- преднамеренное форматирование данных (сдвиг слов, предложений, абзацев или выбор определённых позиций символов),

- использование незадействованных участков магнитных и оптических носителей,
- удаление файловых заголовков-идентификаторов и т.д.

В основном, для методов такого типа характерны низкая пропускная способность, слабая производительность, а также низкая степень скрытности.

По назначению выделяют стеганометоды для собственно сокрытия данных в цифровых объектах и методы для скрытого хранения (или скрытой передачи) данных для защиты авторства на них.

По типу контейнера стеганографические методы различают как методы с контейнерами в форме текста, аудио файла, изображения и видео.

1.4. Практическое применение стеганографии

Далее будут приведены задачи, для которых стеганография будет актуальна:

1) Незаметная передача информации. В сравнении с криптографическими методами (которые тайны, но не скрытны), стеганография применяется в качестве незаметной передачи информации. В этом состоит «классическое практическое применение» стеганографии,
2) Скрытое хранение информации. Здесь стеганография используется скорее не для передачи, а для хранения информации. Причём обнаружение факта наличия этой информации (пусть даже зашифрованной) для пользователя нежелательно. Также очевидно, что данная задача реализуется относительно носителя данных, но не каналов связи. А избыточность на некоторых носителях может быть невероятно большой.
3) Не декларированное хранение информации. Информационные ресурсы осуществляют хранение данных только в одном виде.

Однако, используя стеганографию, можно осуществить хранение данных других форматов.

4) Защита авторского права. Тот случай, когда один знак защищает каждую копию контента. Это, например, касается фотографии. Когда происходит публикация фотографии без разрешения фотографа, он сможет доказать авторство стеганографией. В фотографию вкрапляется информация о дате создания снимка и/или какая-нибудь иная информация, позволяющая «привязать» фотографию к единственному фотоаппарату.
5) Защита подлинности документов. Технология аналогична таковой при защите авторского права. Только стеганография используется не с целью подтверждения авторства, а подтверждает подлинность документа. Документ считается ненастоящим, т.е. поддельным, если не содержит СВЗ.
6) Индивидуальный отпечаток в системе электронного документооборота (СЭДО). Индивидуальный отпечаток в СЭДО можно применить внутри документов при работе человека с ними. Для этого специальные приложения, драйверы, установленные и работающие в системе, должны быть написаны и сконфигурированы. Если этот пункт выполнен, то с помощью индивидуального отпечатка будет возможно опознать, кто работал с документом. Однозначно, стеганографию здесь не следует делать единственным критерием, но в качестве дополнительного способа идентификации участников, работающих с документом, она может быть очень полезна.
7) Водяной знак в DLP-системах (Data Leak Prevention). Стеганография может быть применена для предотвращения утечек информации. В этом направлении стеганография применяется при создании документа, содержащего секретную информацию, вкрапляется некоторая метка. При этом, вне зависимости от количества копий

документа, она не изменяется. Чтобы извлечь метку, нужен стегоключ. Разумеется, он держится в тайне от основной массы пользователей. DLP-система выверяет присутствие или отсутствие водяного знака перед одобрением или отказом соответственно выдать документ вовне. Если знак имеется, то система не позволит отправлять документы вовне.

8) Скрытая передача управляющего сигнала. Пусть получателем выступает какая-либо система, а отправителем будет оператор. Здесь стеганография применяется для доставки управляющего сигнала системе. Если она может менять свои состояния, и мы не хотим, чтобы противник догадался о переходе системы в другое состояние, можно воспользоваться стеганографией. Криптография без стеганографии информирует противника о том, что что-то изменилось и, соответственно, спровоцировать на дальнейшие нежелательные действия.

9) Подтверждение достоверности переданной информации. Тут стегосообщение имеет данные, которые подтверждают корректность передаваемых данных контейнера. Эту роль выполняет хэш-функция или контрольная сумма. Задача подтверждения достоверности будет актуальна, когда у противника есть необходимость подделать данные в контейнере. С защитой подлинности документов это применение путать не стоит. Опять-таки, если говорить о фотографии, защитой подлинности будет доказательство того, что она настоящая. В случае же подтверждения достоверности, нужно установить защиту от третьей стороны, которая обладает способностью к подделке данных между отправителем и получателем. Большое количество классических решений, криптографических в том числе, существует для данной проблемы. Стеганография, как способ, также применятся для решения.

10) Неотчуждаемость информации. Для ряда документов важна целостность. Этого можно достичь с помощью резервирования данных. Проблемы возникают, если нужно хранить документы таким образом, чтобы не было возможности отделить одну информацию от другой информации. Ярким примером являются медицинские снимки. Если читать, что противника не существует, можно вкраплять информацию без ключа. То есть использовать стеганографию только для «неотчуждения» от дополнительной информации самих снимков, понадеявшись на честность пользователей. Вероятно, это удобнее, чем сопровождать каждый снимок информацией.

1.5 Выводы

Данная глава раскрывает суть термина «стеганография», её методы и разновидности. Производится небольшой ввод в историю формирования стеганографии как науки, а также обуславливается происхождение названий некоторых основных её понятий. Описывается её практическое применение к современным реалиям и разрабатывающимся системам.

ГЛАВА 2
МЕТОДЫ ВНЕДРЕНИЯ ЦВЗ

Для погружения в проблему внедрения ЦВЗ в электронные документы необходимо было рассмотреть уже существующие подходы в данной области.

2.1. Вставка младшего бита при малом размере контейнера

Встраивание в менее значимые биты контейнера – это наиболее известный и, исторически, один из первых подход, который может применяться как для защиты сигналов с помощью ЦВЗ, так и для стеганографии. Он позволяет встроить достаточно большое количество информации без скольких-нибудь заметных искажений контейнера, однако методы, использующие данный подход, как правило, обладают низкой стойкостью к искажениям носителя информации и относительно легко могут быть подвергнуты стегоанализу, поэтому имеют весьма ограниченную применимость.

Основная идея метода заключается в том, что любое изображение может быть представлено в виде совокупности битовых плоскостей. Так, контейнер $C(n_1, n_2)$ будет иметь вид:

$$C(n_1, n_2) = C_1(n_1, n_2) + C_2(n_1, n_2) \cdot 2 + \ldots + C_K(n_1, n_2) \cdot 2^{K-1}, \qquad (2.1)$$

где $C_k(n_1, n_2) \in [0,1]$ – битовые плоскости, k – номер битовой плоскости, $K = 8$ – их количество.

Наименее и наиболее значащими битовыми плоскостями являются соответственно C_1 и C_8 : если изменить значение бита $C_1(n_1, n_2)$, то яркость изменится на единицу; если же изменить значение бита $C_8(n_1, n_2)$, то яркость изменится на 128;

Ключевая идея заключается в использовании младших бит в цветовом канале каждого пикселя как места, где спрятана информация.

Всё, что необходимо, это стегосообщение и, непосредственно изображение-контейнер. Работа этого метода производится на битовом уровне, что и влечёт применение такого подхода в качестве способа сокрытия информации.

Беря в учёт размеры изображения (ширина, длина и RGB-слои), мы получаем конечное число пиксельных каналов для сокрытия сообщения. Остаётся лишь подогнать пиксельные каналы стегосообщения под количество таковых у контейнера, и можно осуществить сокрытие.

Также возможен и обратный процесс. Необходимо лишь получить от каждого пиксельного канала его младшие биты и сгруппировать из них слова.

Недостатком этого метода является ограничение на размер стегосообщения. Дело в том, что сокрыть сообщения больших размеров таким образом также можно, но не без деформации контейнера. А это уже не соответствует преследуемой цели незаметно передать информацию.

2.2. Вставка младшего бита при большом размере контейнера

Если размеры контейнера небольшие, то при вставке в него большого количества информации его первоначальный вид наверняка исказится. То есть появится много шумов.

Чтобы спрятать большое сообщение без значительной деформации контейнера, надо выбирать и контейнер больше. Это может повлечь различные проблемы. Если, например, необходимо передать очень большое сообщение, то нужно, чтобы размер контейнера был в несколько раз больше самого контейнера. А уже это может нанести ущерб стеганостойкости, поскольку потенциальный противник первоочерёдно проверит большие изображения. Хотя это можно

обыграть и в свою пользу, отправляя много больших файлов. Таким образом противник будет просто загружен ненужным анализом и работой.

В данной ситуации поможет создание стегосистемы с применением ключа. Необходимо, чтобы ключ неравномерным образом менял координаты последующего пикселя, где будет размещена следующая часть информации.

Таким образом за контейнер в данном подходе можно взять изображение либо текст.

Следует начать с ключа. Ключ будет представлять собой набор групп из трёх координат, где количество групп будет соответствовать длине стегосообщения. Координаты можно дать по двум осям x и y. Но, имея наборы трёх цветов в RGB, нужно ещё и третью ось z, где будет значение от 0 до 2.

Рисунок 2.1 – а) Расширение координат; б) Расширенная ось x после разделения значения RGB по ширине

Объединив разнородные координаты трёх осей, получим одну уникальную координату x. Это делается для уменьшения размера ключа. Одно из значений цвета соответствует значению символа в ASCII таблице. Засчёт этого контейнер может поместить больше информации.

Смысл алгоритма заключается в поиске координаты, значение которой будет соответствовать значению символа.

В случае сокрытия текста в текстовой строке, весь набор слов можно представить в качестве строки. И каждый элемент его будет иметь определённую координату по оси x.

Подход не очень хорош тем, что пользователь сам ключ не придумывает, а его выводит алгоритм. Получается, что новый ключ передаётся при каждой последующей передаче данных, что является плохим тоном для стеганографии.

Несмотря на то, что алгоритм выдаёт ключи, их можно использовать неограниченно. Для этого необходимо взять другой контейнер так, чтобы в заданных координатах каждый раз было другое сообщение. Это можно сделать по-разному: взять за основу изображение в открытом доступе либо с нуля создать такое изображение. Когда контейнер – текст, то действия аналогичны указанным ранее для изображения.

В зависимости от размеров изображения, количество ключей может быть очень большим. Помимо всего прочего, передача сообщения может быть неограниченна одним форматом контейнера либо одним контейнером. Если, к примеру, стегосообщение передаём через блог, то одна часть сообщения может находится в тексте, а другая – в видео.

В одно изображение или большой текст может поместиться бесконечное количество сообщений. Лучше, если изображение является разноцветным, поскольку это обеспечивает многообразие вариантов. В чёрно-белом изображении передать большое сообщение будет тяжело.

В данном подходе контейнер не подвергается деформации: оригинал и используется в качестве контейнера.

2.3. Внедрение бинарных данных в HTML-документы

Невозможно что-либо вставить в HTML документ, т.к. всё вставленное будет либо видимым в браузере, либо видимым в исходном тексте как бесполезный материал. Однако, если обратиться к структуре HTML-документа, то он всегда состоит из тегов и присущих им атрибутов. И если порядок следования тегов нарушить нельзя, так как это приведёт к деформации структуры документа, то порядок атрибутов там можно изменить, не изменяя видимый документ или размер файла.

В классической текстовой стеганографии понадобилось бы 80 строк текста для сокрытия 10 символов секретного текста. К счастью, HTML-документы

имеют более общие комбинации атрибутов, особенно, если использовать старый формат HTML с прописанными в нём стилями CSS.

Смысл заключается в следующем: для каждого тега, имеющего 2 и более атрибутов, выделяется пара его атрибутов, порядок следования которых является истинным, то есть соответствует 1. При замене данных атрибутов местами получается ложная пара, соответствующая 0.

Таким образом, имея бинарное представление сообщения и набор пар атрибутов, можно путём изменения порядка их следования сокрыть это сообщение.

Обратный процесс получения исходного сообщения также не составит труда, если знать, какие именно пары атрибутов использовались для сокрытия и какой порядок следования атрибутов принят за истинный.

Недостатком этого метода является небольшой размер сообщения, которое можно сокрыть.

2.4 Дискретное вейвлет-преобразование

Дискретное вейвлет-преобразование является значимым инструментом, если говорить о воздействии на изображения. Оно основано на одномерном вейвлет-преобразовании, которое не зависит от числа столбцов и строк изображения. Поэтому предпочтение отдаётся горизонтальному и вертикальному направлениям.

Вейвлеты Хаара представляют из себя кусочно-постоянные функции, которые заданы на конечных интервалах различного масштаба и принимают два значения $\{-1; +1\}$. Материнский вейвлет Хаара, или вейвлет Хаара нулевого смещения и единичного масштаба, это функция, равная +1 на промежутке $[0; \frac{1}{2})$ и -1 на промежутке $[\frac{1}{2}; 1)$. В практических заданиях по обработке дискретного сигнала, например, цифровые фотографии или массивы отсчётов аудио-сигналов, вейвлеты Хаара зарекомендовали себя хорошо. Характерная

особенность проебразования Хаара в том, что оно разделимо и легко вычисляется.

2.4.1 Преобразование Хаара

Преобразование Хаара (ПХ) считается одним из простейших базисных вейвлет-преобразований. Пусть имеется одномерный сигнал $f = (f_1, f_2, \dots, f_N)$. ПХ разлагает сигнал на два компонента, где один называется разностью, а другой известен как среднее. У под-сигнала первое среднее значение $a^1 = (a_1, a_2, \dots, a_{N/2})$ на первом уровне для сигнала длины N $(f = (f_1, f_2, \dots, f_N))$ вычисляется по формуле:

$$a_n = \frac{f_{2n-1}+f_{2n}}{\sqrt{2}}, n = 1,2,3, \dots, \frac{N}{2} \quad (2.2)$$

и первый детализирующий под-сигнал $d^1 = (d_1, d_2, \dots, d_{N/2})$ на этом же уровне представляется в качестве:

$$d_n = \frac{f_{2n-1}-f_{2n}}{\sqrt{2}}, n = 1,2,3, \dots, \frac{N}{2} \quad (2.3)$$

Данные значения порождают два новых сигнала: $a = \{a_n\},\ n \in Z$ и $d = \{d_n\},\ n \in Z$, где один является огрубленной версией исходного (каждой паре элементов f ставится в соответствие их среднее арифметическое), другой же содержит детализацию, нужную для восстановления исходного сигнала. Действительно:

$$f_{2n-1} = a_n + d_n, \quad f_{2n} = a_n - d_n, \ \ n \in Z \quad (2.4)$$

Можно применить аналогичную операцию к сигналу a, получив таким образом два сигнала, среди которых один будет огрубленной версией a, другой содержит необходимую для восстановления a детализирующую информацию.

Быстрое преобразование Хаара включает вычитание, сложение и деление на 2, вследствие чего оно является эффективным и сводит вычислительную

задачу к сравнению с ПХ. Чтобы разложить изображение, одномерное ПХ сначала следует применить к каждой строке пиксельных значений отображающей матрицы, а после уже и к каждому столбцу. Все детализирующие коэффициенты являются результирующими значениями, за исключением отдельного общего среднего коэффициента.

2.4.2 Двумерное вейвлет-преобразование

Двумерное вейвлет-преобразование матрицы подразумевает поочерёдное одномерное вейвлет-преобразование строк и столбцов данной матрицы. Вначале для каждой строки производится одномерное вейвлет-преобразование, после чего преобразованная строка записывается на прежнем месте. Аналогичным образом вейвлет-преобразование применяется ко всем столбцам. Как результат, изображение разбивается на четыре равные части:

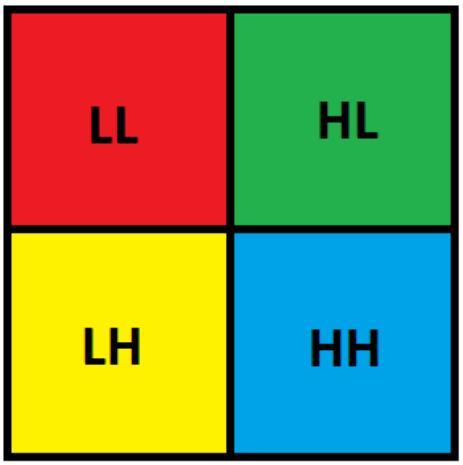

Рисунок 2.2 Пример воздействия двумерного вейвлет-преобразования на изображение квадратной формы

N-кратное двумерное вейвлет-преобразование подразумевает применение двумерного вейвлет-преобразования N раз, причём каждое последующее двумерное вейвлет-преобразование применяется к младшей четверти матрицы:

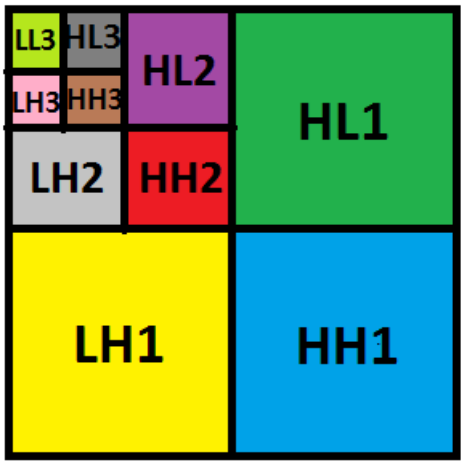

Рисунок 2.3 Применение двумерного вейвлет-преобразования последовательно 3 раза к изображению квадратной формы

Обратное двумерное вейвлет-преобразование же рекурсивно восстанавливает младший квадрант матриц. Указанное преобразование является иерархическим. Иными словами, если при использовании обратного вейвлет-преобразования вычисляются не все уровни, а меньшее их количество, то в квадранте LL образуется уменьшенная копия изображения. Если не предполагается применение обратного вейвлет-преобразования, то младший квадрант, таким образом, становится уменьшенной копией изображения. Обратное вейвлет-преобразование, отсюда, позволяет вырезать фрагменты изображений различного масштаба. Однако доступные масштабы определяются количеством уровней вейвлет-преобразования и масштабы непроизвольны, а увеличены в два раза.

2.5 Выводы

Данная глава даёт подробное описание с теоретической точки зрения таким методам как вставка младшего бита, внедрение бинарных данных в HTML-документ и преобразование Хаара; раскрываются их преимущества и недостатки.

ГЛАВА 3
РЕАЛИЗАЦИЯ МЕТОДОВ СОКРЫТИЯ ДАННЫХ

3.1 Метод вставки младшего бита при малом размере контейнера

В роли контейнера было взято произвольное изображение размера 255×255 пикселей:

Рисунок 3.1 Изображение-контейнер

Для реализации в качестве программного средства автором использовалась вычислительная среда Wolfram Mathematica 11.0. Некоторые особенности данной среды также учитывались при написании кода и о них речь пойдёт ниже.

Во-первых, Mathematica работает в очень точном цветовом пространстве, поэтому вычислительно будет сложно работать с однобайтными цветовыми каналами. Вначале получается битовое содержимое контейнера, которое затем каналы которого приводятся к 8-битному состоянию, как показано на изображении ниже:

Рисунок 3.2 Изображение с 8-битными каналами

После подготовки стегоконтейнера переходим к сообщению, которое хотим передать. В качестве демонстрации было взято сообщение «Secret message».

Поскольку работа ведётся на битовом уровне, необходимо также получить битовое представление данного стегосообщения. Далее возникает вопрос о способе вставки сообщения в контейнер, если количество их битовых каналов разнится. Самым простым способом разрешить этот вопрос является дополнение недостающих каналов нулями до параметров изображения:

$$\text{ширина} * \text{длина} * 3 \text{ слоя } RGB \tag{3.1}$$

Всё, что остаётся сделать, так это сложить битовое представление контейнера и сообщения. Визуально изображение со вставленным сообщением будет выглядеть так:

Рисунок 3.3 Изображение со стегосообщением

Поставленную задачу этот метод выполнил. Визуально человек не обнаруживает разницу.

Чтобы всё-таки подтвердить тот факт, что с контейнером произошло изменение, вычтем из исходного изображения [Рисунок 3.1] изображение [Рисунок 3.3]:

Рисунок 3.4 Результат вычитания изображения со стегосообщением из исходного изображения

Разноцветные участки на изображении и подтверждают факт произошедших изменений с контейнером. Разница лишь в том, что одни цветные участки являются результатом приведения изображения к 8-битным пиксельным каналам, другие – вставки стегосообщения, а третьи – комбинации этих двух операций.

Для извлечения сообщения обратно нужно собрать младшие биты из всех битовых каналов, соединить их в 8-битные слова и сконвертировать к текстовым символам.

Таким образом на выходе опять получим «Secret message».

3.2 Метод вставки младшего бита при большом размере контейнера

Реализация метода осуществлена в виде небольшой программы:

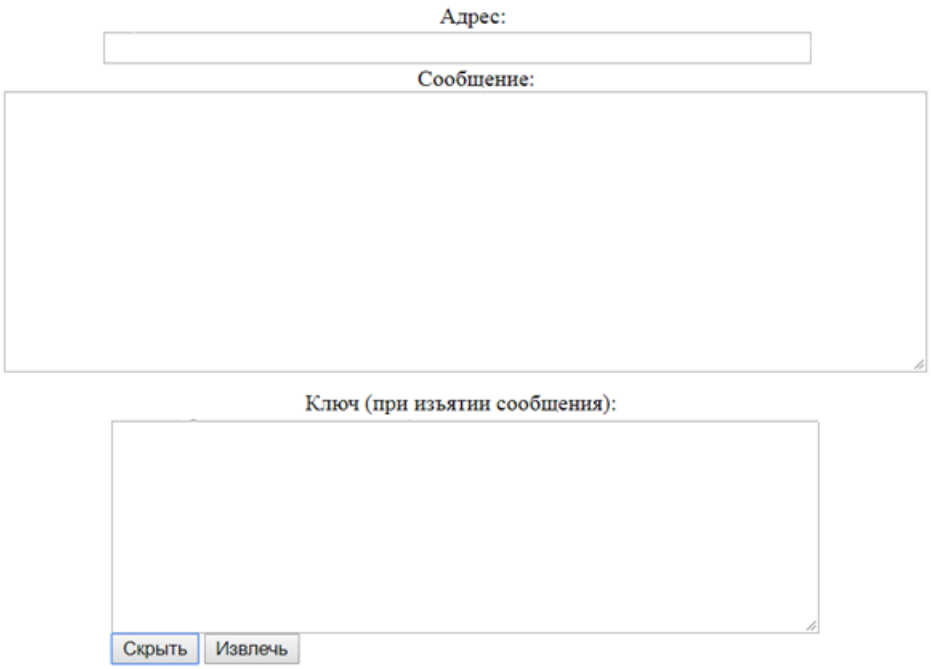

Рисунок 3.5 Внешний вид реализованной программы

Данный вариант программы был реализован для текстового контейнера.

Всё начинается с ввода текста-контейнера и самого стегосообщения. Необходимым условием для простоты вычислений было введение ограничений на длину стегосообщения не более 100 символов.

Рисунок 3.6 Фрагмент программы с введённым текстом-контейнером и стегосообщением

Далее проверяется текст-контейнер на наличие в себе символов из стегосообщения. Если символ найден, то его позиции в стегосообщении ставится в соответствие один из символов a-z, что соответствует 36 системе счисления в

языке PHP. Если же символ не найден, то его позиция переводится в ASCII таблицу. По такому принципу формируется вышеупомянутый ключ.

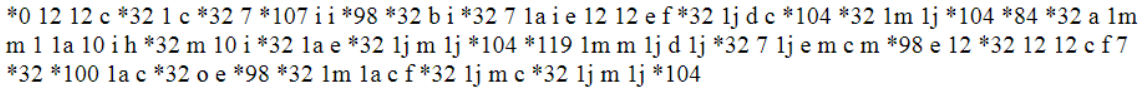

Рисунок 3.7 Сгенерированный ключ

Чтобы получить сообщение обратно по ключу и контейнеру, необходимо произвести все преобразования в системах счисления в обратную сторону.

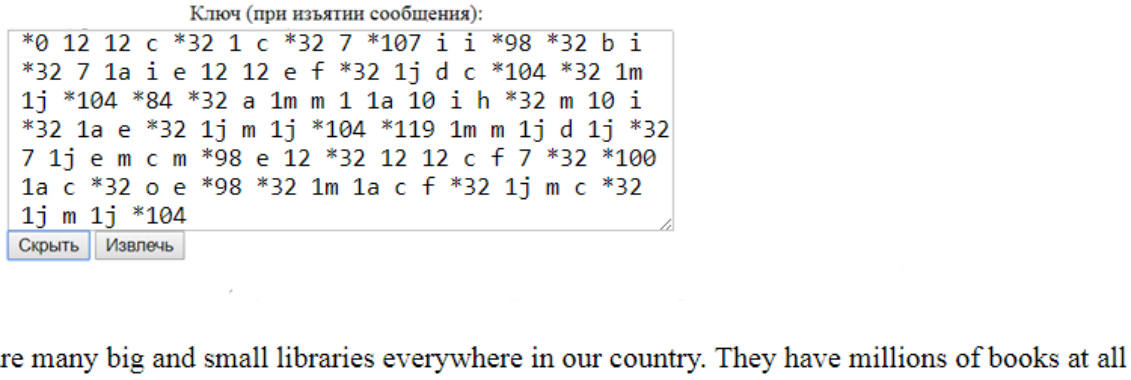

Рисунок 3.8 Фрагмент программы с восстановленным сообщением.

3.3 Внедрение бинарных данных в HTML-документ

Рассмотрим следующий HTML-документ:

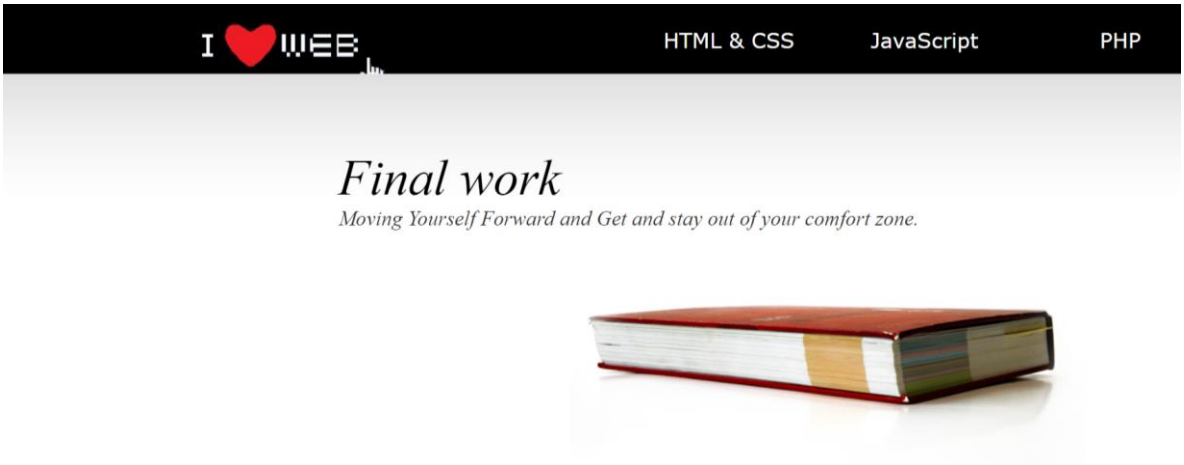

Рисунок 3.9 Внешний вид HTML-документа

Как упоминалось ранее, структуру HTML-документа изменить нельзя, так как это приведёт к его непосредственной деформации. Поэтому перейдём на структурный уровень тегов и атрибутов.

Теги <meta>, <link>, , <ul>, <a>, <img> – самые часто встречающиеся в структуре документа и имеют 2 и более тегов. Структура документа приведена в Приложении В.

Выделим пары атрибутов, которые будем считать истинными:

Атрибут 1	Атрибут 2
http-equiv	content
type	rel
href	name
class	id
height	width

Таблица 3.1 Истинные пары атрибутов

Далее необходимо сообщение, которое нужно сокрыть. Как уже упоминалось, этот подход не позволяет скрывать большие сообщения. Поэтому в качестве примера берём сообщение: «no».

Его вид согласно таблицы ASCII будет выглядеть как «110 111 033», а в бинарном виде «01101110 01101111».

Переходим непосредственно к процессу сокрытия. Первая строка с подходящей нам парой: <meta http-equiv="Content-language" content="ru" />. Как видно, она стоит в том порядке, как и в таблице, то есть истинном. Однако согласно бинарного представления сообщения, первым должен быть 0, а не 1. Следовательно производим замену атрибутов местами: <meta content="ru" http-equiv="Content-language"/>. Вторая строка с подходящей парой: <meta http-equiv="Content-Type" content="text/html; charset=utf-8">. Здесь также атрибуты стоят в том порядке, в каком указано в таблице. Но поскольку необходимо

сокрыть 1, то порядок атрибутов не меняем. Аналогично поступаем для следующей строки: <link href="css/stylesheet.css" rel="stylesheet" type="text/css" />. Тут атрибуты не стоят в порядке, определённым в таблице. Значит такая пара являет собой 0. Однако спрятать необходимо 1, поэтому вновь меняем местами атрибуты: <link href="css/stylesheet.css" type="text/css" rel="stylesheet" />.

Продолжаем эту процедуру, пока не сокроем всё сообщение. Теперь, после того как все атрибуты стоят на местах согласно сокрытого сообщения, можно взглянуть на HTML-документ снова:

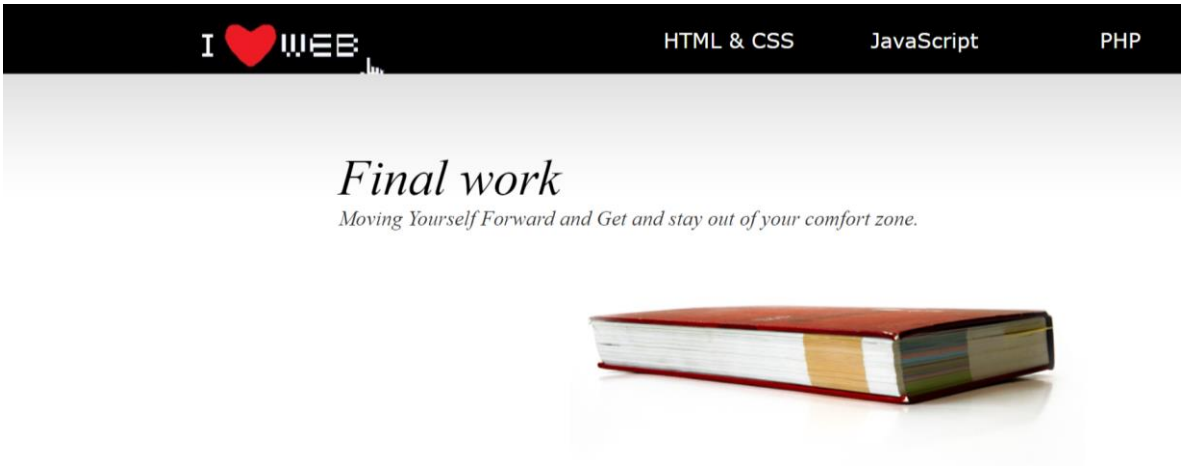

Рисунок 3.10 Внешний вид HTML-документа после внедрения сообщения

Как видно, ничего не изменилось, за исключением внутренней организации атрибутов в документе.

Для восстановления же исходного сообщения необходимо просто проитерироваться по всем парам и сравнить положение их атрибутов с таковыми в таблице, получая при этом значения 0 и 1. А зная, что сообщение было сокрыто в бинарной форме, не составит особого труда полученный набор единиц и нулей конвертировать обратно в текст. Поэтому на выходе получим снова сообщение: «no».

3.4 Внедрение водяного знака в PDF-документ

Используя рассмотренное ранее двумерное вейвлет-преобразование изображения, можно придать ему практическое применение для рассматриваемой темы в дипломной работе.

Довольно часто можно наблюдать, что PDF-документы помечены определёнными изображениями, либо на протяжении всей страницы там размещена расплывчатая надпись с указанием владельца документа. Это тоже водяной знак.

Как уже говорилось, вейвлет-преобразование можно последовательно применить к изображению несколько раз, то есть совершить многомерное вейвлет-преобразование. Для изображения это чревато тем, что его можно масштабировать до сколь угодных размеров, вплоть до размеров 1× 1 px. Не стоит забывать и об обратном вейвлет-преобразовании, восстанавливающем изображение.

Следует отметить, что процедуру прямого вейвлет-преобразования в дальнейшем будем называть «анализом», а процесс обратного вейвлет-преобразования – «синтезом».

Реализация будет заключаться в том, что, выбрав некоторое изображение квадратной формы, можно сжать его до малых размеров и подставить в PDF-документ. Программа-обработчик, получив некоторое изображение, восстанавливает его с помощью синтеза до грублённых очертаний. Таким образом, если это изображение является водяным знаком, то документ деформируется (в реализации автора дипломной работы меняется кодировка документа).

Во-первых, начинаем с анализа изображения:

Рисунок 3.11 Изображение, которое будет использоваться в качестве водяного знака

В целях повышения наглядности изображение будет сжиматься не до размеров 1×1 px, а до размеров 10×10 px.

Осуществляем процесс анализа. Визуально он выглядит так:

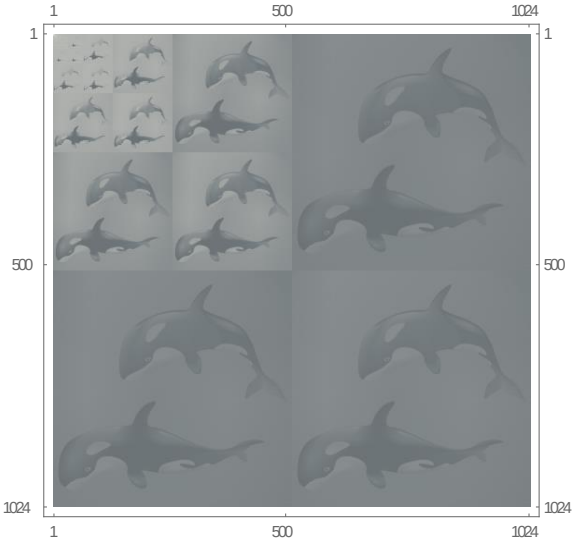

Рисунок 3.12 Анализ изображения

Осуществив данную процедуру, можем выделить изображение необходимого нам размера 10×10 px. Визуальная разница очевидна:

Рисунок 3.13 а) Изображение исходное; б) Изображение размера 10×10 px

Далее нам необходим PDF-документ, непосредственно в который будет внедряться водяной знак. Для этих целей был взят простой фрагмент некоторого текста.

Внедрение водяного знака, его извлечение, а также процедуры анализа и синтезабыли реализованы в написанной автором программе:

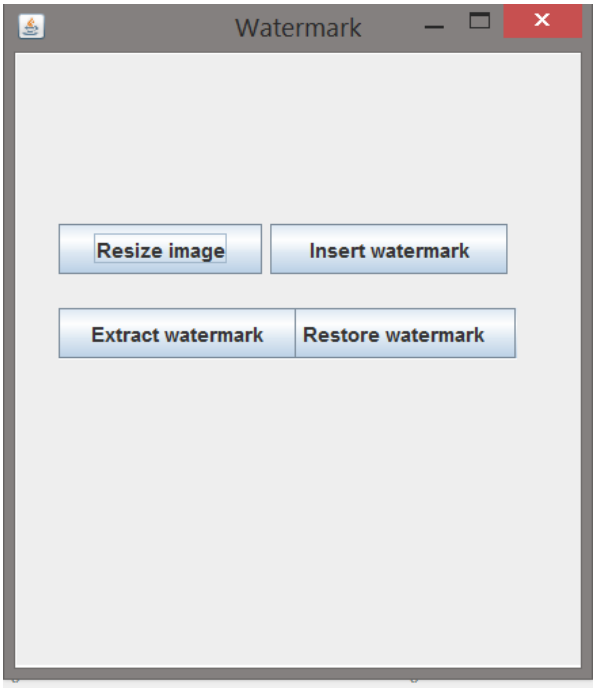

Рисунок 3.14 Интерфес программы

Следующим шагом идёт внедрение водяного знака в PDF-документ. Используя программные средства языка Java, это выглядит так:

Поклонилась ему в ноги дочь середняя и говорит:

- Государь ты мой батюшка родимый! Не вози ты мне золотой и серебряной парчи, ни чёрных мехов соболя сибирского, ни ожерелья жемчуга бурмицкого, ни золота венца самоцветного, а привези ты мне тувалет из хрусталю восточного, цельного, беспорочного, чтобы, глядя в него, видела я всю красоту поднебесную и чтоб, смотрясь в него, я не старилась и красота б моя девичья прибавлялася.

Призадумался честной купец и, подумав мало ли, много ли времени, говорит ей таковые слова:

- Хорошо, дочь моя милая, хорошая и пригожая, достану я тебе таковой хрустальный тувалет; а и есть он у дочери короля персидского, молодой королевишны, красоты несказанной,

Рисунок 3.15 Фрагмент PDF-документа с водяным знаком

При извлечении изображения происходит процедура синтеза и восстановления изоюражения, воспринятого как водяной знак до его исходных размеров:

Рисунок 3.16 Изображение после процедуры синтеза

Поскольку изображение является водяным знаком, разрушаем документ (меняем кодировку в тексте):

- ???????? ?? ??? ??????? ????????! ?? ???? ?? ??? ??????? ? ?????????? ?????, ?? ?????? ????? ?????? ??????????, ?? ???????? ??????? ??????????, ?? ?????? ????? ????????????, ? ??????? ?? ??? ??????? ?? ???????? ??????????, ????????, ????????????, ?????, ????? ? ????, ?????? ? ??? ??????? ??????????? ? ????, ???????? ? ????, ? ?? ????????? ? ??????? ? ??? ??????? ????????????.

???????????? ??????? ????? ?, ??????? ???? ??, ????? ?? ???????, ??????? ?? ??????? ?????:

Рисунок 3.17 PDF-документ после деформации

Рассмотренный процесс является упрощённой версией реально происходящего процесса. Как изображение для вставки, так и сама его вставка могут быть усложнены дополнительными методами.

3.5 Выводы

Данная глава содержит подробную практическую реализацию автора дипломной работы рассмотренных в предыдущей главе методов; приведены промежуточные и конечные результаты работы программ и методов.

ЗАКЛЮЧЕНИЕ

Несмотря на своё относительно недавнее формирование, стеганография хорошо зарекомендовала себя на поприще сокрытия данных и защиты авторских прав. Она касается буквально каждой сферы жизни человека и используется повсеместно.

Стеганографические методы и подходы постоянно совершенствуются и дополняются, что позиционирует их как одно из популярных направлений для развития и разработок в современной науке.

В ходе дипломной работы её автором были изучены и реализованы методы вставки младшего бита для контейнеров большого и малого размеров, метод сокрытия бинарных данных в HTML-документе, внедрение водяного знака в PDF-документ.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1) Федосеев, В.А. Цифровые водяные знаки и стеганография: учеб. пособие с заданиями для практ. и лаб. работ. – Самара, Издательство СГАУ, 2015 – 128 с.
2) Изычева, А.В. Стеганографические методы защиты информации: учеб. пособие / А.В. Изычева, В.Г. Сидоренко. – Москва: РУТ (МИИТ), 2017. – 76 с.
3) Сунчугашев, И.В. Стеганография: учеб. пособие. – Москва: ГУ МФТИ, 2008. – 14 с.
4) Абазина, Е.С. Цифровая стеганография: состояние и перспективы / Научный журнал «Системы управления, связи и безопасности». – Санкт-Петербург: БГТУ, 2016. – 20 с.
5) Технические статьи [Электронный ресурс] / The code project, 2012. – Режим доступа: https://www.codeproject.com/. – Дата доступа: 20.03.18
6) Буй, Т. Т. Ч. Разложение изображений с помощью двумерного дискретного вейвлет-преобразования и быстрого преобразования Хаара: учеб. пособие / Т. Т. Ч. Буй, В. Г. Спицын. – Томск: ТПУ, 2011. – 5 с.
7) Федосеев, В. А. Унифицированная модель систем встраивания информации в цифровые сигналы: научная статья / В. А. Федосеев. – Самара: СГАУ, 2016. – 12 с.
8) Лекционные материалы Студопедия [Электронный ресурс] / Studopedia, 2013. – Режим доступа: https://studopedia.su/. – Дата доступа: 12.04.2018
9) IT – проект SavePearlHarbor [Электронный ресурс] / Тематические медиа, 2006. – Режим доступа: http://savepearlharbor.com/. – Дата доступа: 01.05.2018
10) Митрофанова, Е. Ю. Нейросетевые сжимающие преобразования данных и алгоритмы создания цифровых водяных знаков в объектах мультимедиа

графических и звуковых форматов: диссертация / Е. Ю. Митрофанова. – Воронеж: ВГУ, 2014. – 179 с.

11) Грачёва, Ю.А. Применение цифровых водяных знаков для защиты цифровых фотографий: научная статья / Ю.А. Грачёва. – Москва: МГУП, 2013. – 6 с.
12) Генне, О.В. Основные положения стеганографии: Научный журнал «Защита информации. Конфидент» / О.В. Генне. – Санкт-Петербург: РИНЦ, 2005. – 5 с.

ПРИЛОЖЕНИЕ А.

Вставка младшего бита при малом размере контейнера

```
hideMsg[img_, msg_] := Block[{imgData, trunkateImg, pixelCell, secretBit},
  imgData = ImageData[img, "Byte"]; pixelCell = Apply[Times,
Dimensions[imgData]]; secretBit =
PadRight[Flatten[IntegerDigits[ToCharacterCode[msg], 2, 8]], pixelCell];
  trunkateImg = BitAnd[imgData, 2^^11111110];
  Image[truncateImg + Fold[Partition, secretBit,
Reverse@Rest[Dimensions[imgData]]], "Byte"]]


takeMsg[img_] := Block[{secretMsg},
  secretMsg = BitAnd[ImageData[img, "Byte"], 1];
  secretMsg = Partition[Flatten[secretMsg], 8];
  secretMsg = FromCharacterCode[FromDigits[#, 2] & /@ secretMsg];
  StringTrim[secretMsg, FromCharacterCode[{0, 0, 0, 0, 0, 0, 0, 0}] ..]]
```

ПРИЛОЖЕНИЕ Б.

Вставка младшего бита при большом размере контейнера

```php
<?php

        $lengthText = strlen($_POST["text"]);
        $length_cont = strlen($_POST["cont"]);
        $text = $_POST["text"];
        $text_cont = $_POST["cont"];

        $code36 = "";
        while ($lengthText) {
                $coord = strpos($text_cont, $text[$length]);
                if ($coord)
        { $code36.= base_convert ($coord, 10, 36)." "; }
                        else
        { $code36.= " *".ord($text[$lengthText])." "; }
                $lengthText--;
        }
        echo "Ключ:<br>".$code36;
?>
```

```php
<?php
        $cont = $_POST["cont"];
        $length = count($key)+1;
        $key = explode(" ",$_POST["key"]);
        $text="";
        while ($length--) {
                $k_coord = $key[$length];
                if ($k_coord[0]!="*") {
                        $coord10 = base_convert($k_coord, 36, 10);
                        $text.= $cont[$coord10];
                } else {
                        $k_coord = str_replace("*","",$k_coord);
                        $text.= chr($k_coord);
                }}

        echo $text;
?>
```

ПРИЛОЖЕНИЕ В.

Сокрытие бинарных данных в HTML-документе

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-language" content="ru" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<link href="css/stylesheet.css" rel="stylesheet" type="text/css" />
<link rel="shotrcut icon" href="picture_empty_1216.png" type="image/png">
<title>Финальный сайт</title>
</head>

<header>
    <div id="topblack">
        <div id="header_container">
          <div id="header_image"> </div>
            <div id="nav_block">

                <div class="nav_button">

                        <ul class="navigation">

                        <li>HTML & CSS
                            <ul>
                              <li><a href="file:///D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A3%20%B8%D0%B51.html">work 1</a></li>
                                    <li><a href="file:///D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%B8%D0%B52-3.html">work 2</a></li>
                                    <li><a href="file:///D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A3%20%D0%BE%D1%82%D1%8B/3/3.html">work 3</a></li>
                              <li><a href="file:///D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A3%20%D0%9C%D0%9C%D0%A4%20%D0%BE%D1%82%D1%8B/4/4.html">work 4</a></li>
                                    <li><a href="file:///D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A3%20%D0%9C%D0%9C%D0%A4%82%D1%8B/5%28CSS%29/5.html">work 5</a></li>
                                     <li><a href="file:////D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A9%20%D0%95%D0%95.html.html">work 6</a></li>
                            </ul>
                            </li>

                            <li>Javascript
                                <ul>
                                  <li><a href="file:////D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%93%D0%A3%20%D0%D1%82%D1%8B/JavaScript/1/new%201.html">work 1</a></li>
                                        <li><a href="file:////D:/%D0%9C%D0%B0%D1%80%D0%B8%D1%8F/%D0%91%D0%%B1%D0%BE%D1%82%D1%8B/JavaScript/2/new%202.html">work 2</a></li>
                            </ul>
                            </li>
                        <li>PHP
                            <ul>
                                <li><a href="http://php.ru/laba%201.php">work 1</a></li>
                                    <li><a href="http://php.ru/laba2.php?a=5&b=10&n=5">work 2</a></li>
                                    <li><a href="http://php.ru/laba%203.php">work 3</a></li>
                                <li><a href="http://php.ru/laba%205.php">work 4</a></li>
                            </ul>
```

```
Reader r = new FileReader(file);
try {
    BufferedReader br = new BufferedReader(r);
    try {
        String line;

        int line = 0;
        while ((line = br.readLine()) != null) {
            line++;

            String[] tokens = line.split("\\s+");
            int[] values = new int[tokens.length];

            for (int i = 0; i < tokens.length; i++) {
                values[i] = Integer.parseInt(tokens[i]);
            }

            for (int i = 0; i < values.length; i++) {
                for (int j = i + 1; j < values.length; j++) {
                    Pair pair = new Pair(values[i], values[j]);

                    Collection<Integer> lineNumbers;
                    if (lineNumbersByPair.containsKey(pair)) {
                        line = lineNumbersByPair.get(pair);
                    } else {
                        lineNumbers = new HashSet<Integer>();
                        lineNumbersByPair.put(pair, lineNumbers);
                    }
                    lineNumbers.add(line);
                }
            }
        }
    } finally {
        bufferedReader.close();
    }
} finally {
    reader.close();
}
```

ПРИЛОЖЕНИЕ Г.

Преобразование Хаара

```
ClearAll[P, Q, n];
P = {1, 1} / Sqrt[2.];
Q = 1 / Sqrt[2.] Reverse[{1, 1}];
n = Length[P];
armatr[ar_, num_] :=
 Transpose[RotateRight[PadRight[ar, num], 2 * #] & /@ Range[0, (num - 1)/2]]
arsynt[arP_, arQ_, num_] := With[{tempP = armatr[arP, num], tempQ = ar2matr[arQ, num]},
  Flatten[Append[tempP[[#]], tempQ[[#]]] & /@ Range[Length[tempP]]
 ]
syntAr = arsynt[P, Q, 2^#] & /@ Range[1, 10];
syntFunMatr[pos_: Array, level_: 1] := Block[{wlGram},
   wlGram = pos;
   (wlGram[[ ;; Length[#]]] = #.wlGram[[ ;; Length[#]]];
      wlGram = Transpose@wlGram;
      wlGram[[ ;; Length[#]]] = #.wlGram[[ ;; Length[#]]];
      wlGram = Transpose@wlGram;
     ) & /@ syntAr[level ;;];
   wlGram
  ];
```

```
ClearAll[analFun];
analizeFun[signAr_, level_: 1, norm_: True] := Block[{wlGram},
 wlGram = signAr;
 (wlGram[[;; Length[#]]] = #.wlGram[[;; Length[#]]]) & /@ Reverse[analizeAr][[level ;;]];
 If[norm, Normalize[#] & /@ wlGram, wlGram]
 ]
analizeStep2D[sign_, analizeAr_, norm_: True] := Block[{wlGram},
 wlGram = sign;
 wlGram = analizeAr.wlGram;
 wlGram = Transpose@wlGram;
 wlGram = analizeAr.wlGram;
 wlGram = Transpose@wlGram;
 wlGram = If[norm, Rescale@wlGram, wlGram]
 ]
analizeAll = Transpose[#] & /@ syntAr;
analizeFun2D[signAr_, level_: 1, norm_: True, analizeAr_: analizeAll] := Block[{wlGram, tempAr, tempLen},
 wlGram = {signAr};
 (tempLen = Length[#];
   tempAr = Last[wlGram];
   tempAr = tempAr[[;; tempLen, ;; tempLen]];
   AppendTo[wlGram, analStep2D[tempAr, #, norm]]
   ) & /@ Reverse[analizeAr][[level ;;]];
 Rescale@wlGram
 ]
```

Внедрение водяного знака в PDF-документ

```
public static void restoreFunction() throws IOException, DocumentException {
        BufferedImage image = ImageIO.read(input);

        BufferedImage resized = resize(image, 1024, 1024);

        File output = new File(RESIZED_PICTURE_NAME);
        ImageIO.write(resized, PICTURE_FORMAT, output);*/
        Document document = new Document();
        PdfWriter.getInstance(document, new FileOutputStream("c://Users/Lisa/Desktop/Watermarking/destroyed.pdf"));
        document.open();
        BufferedReader br = new BufferedReader(new FileReader("c://Users/Lisa/Desktop/Watermarking/preface.txt"));
        String line;
        Paragraph p;
        com.itextpdf.text.Font normal = new com.itextpdf.text.Font(com.itextpdf.text.Font.FontFamily.TIMES_ROMAN, 12);
        com.itextpdf.text.Font bold = new com.itextpdf.text.Font(com.itextpdf.text.Font.FontFamily.TIMES_ROMAN, 12, com.itextpdf.text.Font.BOLD);
        boolean title = true;
        while ((line = br.readLine()) != null) {
            p = new Paragraph(line, title ? bold : normal);
            p.setAlignment(Element.ALIGN_JUSTIFIED);
            title = line.isEmpty();
            document.add(p);
        }
        document.close();
    }

    private static BufferedImage resize(BufferedImage img, int height, int width) {
        Image tmp = img.getScaledInstance(width, height, Image.SCALE_SMOOTH);
        BufferedImage resized = new BufferedImage(width, height, BufferedImage.TYPE_INT_ARGB);
        Graphics2D g2d = resized.createGraphics();
        g2d.drawImage(tmp, 0, 0, null);
        g2d.dispose();
        return resized;
    }
```

```
public static double [] numbCompress(int width, int height, double [][] kernel,
                    double [][] input, int kernelWidth, int kernelHeight){
            double [][] finalRes = new double [width][height];
            finalRes = convolution2DPadded(input,width,height,
                    kernel,kernelWidth,kernelHeight);
            double [] tmp = new double  [width * height];
            for(int j=0;j<height;++j){
                for(int i=0;i<width;++i){
                    tmp[j*width +i]=finalRes[i][j];
                }
            }
            return tmp;
        }

public static int [] greyLvl (double [] color){
            int [] result = new int [color.length];
            int grey;
            for(int i=0;i<color.length;++i){
                if(color[i]>255){
                    grey = 255;
                }else if(color[i]<0){
                    grey = 0;
                }else{
                    grey = (int) Math.round(color[i]);
                }
                result[i] = (new Color(grey,grey,grey)).getRGB();
            }
            return result;
        }
```