Оглавление

[Введение 3](#_Toc513633001)

[1 Анализ предметной области 5](#_Toc513633002)

[1.1 Краткая историческая информация о стеганографии 5](#_Toc513633003)

[1.2 Литературный обзор по теме 7](#_Toc513633004)

[1.2.1 Классическая стеганография 7](#_Toc513633005)

[1.2.2 Компьютерная стеганография 8](#_Toc513633006)

[1.2.3 Цифровая стеганография 8](#_Toc513633007)

[1.3 Атаки на стегосистемы 12](#_Toc513633008)

[1.4 Обзор существующих программных продуктов 13](#_Toc513633009)

[1.5 Патентный поиск по теме проекта 17](#_Toc513633010)

[1.6 Вывод по разделу 20](#_Toc513633011)

[2 Анализ методов текстовой стеганографии 21](#_Toc513633012)

[2.1 Классификация стегосистем 21](#_Toc513633013)

[2.2 Классификация методов цифровой стеганографии 22](#_Toc513633014)

[2.3 Разновидности методов текстовой стеганографии 23](#_Toc513633015)

[2.4 Обоснование пригодности использования XML контейнеров в стеганографии 25](#_Toc513633016)

[2.5 Методы текстовой стеганографии, используемые для внедрения информации в документы типа XML 27](#_Toc513633017)

[2.6 Описание метода текстовой стеганографии на основе замены типа кавычки в XML-документе 28](#_Toc513633018)

[2.7 Использование формата DOCX в качестве стеганографического контейнера 29](#_Toc513633019)

[2.8 Вывод по разделу 30](#_Toc513633020)

[3 Выбор платформы и реализация программного средства 31](#_Toc513633021)

[3.1 Технические требования для реализуемого приложения 31](#_Toc513633022)

[3.2 Выбор технологий разработки 31](#_Toc513633023)

[3.3 Выбор IDE для разработки 36](#_Toc513633024)

[3.4 Разработка приложения 38](#_Toc513633025)

[3.4.1 Разработка алгоритма осаждения/извлечения информации в текстовый контейнер формата DOCX 38](#_Toc513633026)

[3.4.2 Разработка алгоритма внедрения информации 39](#_Toc513633027)

[3.4.3 Разработка алгоритма извлечения информации 45](#_Toc513633028)

[3.4.4 Разработка клиентской части приложения 47](#_Toc513633029)

[3.4 Вывод по разделу 48](#_Toc513633030)

[4 Тестирование программного средства 49](#_Toc513633031)

[4.1 Дымовое тестирование 49](#_Toc513633032)

[4.2 Модульное тестирование 53](#_Toc513633033)

[4.3 Вывод по разделу 57](#_Toc513633034)

[5 Руководство пользователя 58](#_Toc513633035)

[5.1 Активация IIS сервера на операционной системе Windows 58](#_Toc513633036)

[5.2 Разворачивание приложения на IIS 59](#_Toc513633037)

[5.3 Последовательность действий пользователя 62](#_Toc513633038)

[5.4 Вывод по разделу 68](#_Toc513633039)

[Заключение 69](#_Toc513633040)

Введение

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

3

*ДП 00.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Жигаровская С. А.

Утверд.

*Смелов В.В.*

*Введение*

Лит.

Листов

2

БГТУ 84419021, 2018

*419001, 2015*

У

4

Информация всегда играла чрезвычайно важную роль в жизни человека. С давних времен сбор и систематизация сведений об окружающем мире помогали человеку выживать в нелегких условиях — из поколения в поколение передавался опыт и навыки изготовления орудий охоты и труда, создания одежды и лекарств. Информация постоянно обновлялась и дополнялась - каждое изученное явление позволяло перейти к чему-то новому, более сложному. Со временем, большие объемы данных об окружающем мире поспособствовали развитию научно-технического прогресса и, как следствие, всего общества в целом – человек смог научиться управлять различными видами вещества и энергии.

Информация стала одним из важнейших стратегических, управленческих ресурсов, наряду с ресурсами – человеческим, финансовым, материальным. Ее производство и потребление составляют необходимую основу эффективного функционирования и развития различных сфер общественной жизни. А это означает, что не только каждому человеку становятся доступными источники информации в любой части планеты, но и генерируемая им новая информация становится достоянием всего человечества. В современных условиях право на информацию и доступ к ней имеют жизненную ценность для всех членов общества.

С возрастанием значимости информации стало очевидным, что ее передача в открытом виде крайне нежелательна, что породило ряд методов, препятствующих ее транспортировке в открытом виде, а соответственно и защите от сторонних лиц.

За время существования компьютеров уже сложились некоторые методы эффективного противостояния несанкционированному доступу к информации одними из которых является криптография и стеганография.

Криптография – наука о методах обеспечения конфиденциальности (невозможности прочтения информации посторонним) и аутентичности (целостности и подлинности авторства, а также невозможности отказа от авторства) информации посредством шифрования данных.

Изначально криптография изучала методы шифрования информации — обратимого преобразования открытого (исходного) текста на основе секретного алгоритма и/или ключа в шифрованный текст. Традиционная криптография образует раздел симметричных криптосистем, в которых зашифрование и расшифрование проводится с использованием одного и того же секретного ключа.

Помимо этого раздела современная криптография включает в себя асимметричные криптосистемы, системы электронной цифровой подписи (ЭЦП), хеш-функции, управление ключами, получение скрытой информации, квантовую криптографию.

Стеганография – это наука о способах передачи (хранения) сокрытой информации, при которых скрытый канал организуется на базе и внутри открытого канала с применением особенностей восприятия информации.  Главное отличие стеганографии от криптографии состоит в том, что скрывается не только сообщение, но и информация о передаче сообщения. Также стеганография существенно отличается от криптографии тем, что при стеганографических методах шифрования сообщение встраивается в изображение, аудиозапись или текст, то есть в объект, который не привлекает внимание злоумышленников.

При использовании стеганографических методов защита информации происходит за счет выполнения трех условий:

* сокрытие факта передачи скрытой информации;
* сокрытие алгоритма осаждения данных в контейнер;
* сокрытие способа кодирования данных.

Учитывая факт развития вычислительной техники, а также частом использовании таких контейнеров как: аудиозапись, видеофайл, текст, изображение стеганография стремительно набирает популярность среди остальных методов защиты информации. Особый интерес к ней был вызван после того, как в ряде стран были введены ограничения на использование криптосистем.

Целью выполнения дипломного проекта является разработка стеганографического метода хранения конфиденциальной информации в электронных документах на основе использования особенностей языков разметки, а также создание реализующего метод программного средства.

Задачи для достижения цели:

* провести анализ существующих методов осаждения/извлечения информации в электронные документы на основе языков разметки (XML), а также в текстовых документах-контейнерах; выявить недостатки существующих методов, влияющие на объем и целостность осаждаемой информации;
* разработать новый метод текстовой стеганографии на основе использования особенностей языков разметки;
* создать программное средство, реализующее разработанный метод.

1 Анализ предметной области

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

5

*ДП 01.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

*Смелов В.В.*

*Анализ предметной области*

Лит.

Листов

16

БГТУ 84419021, 2018

*419001, 2015*

У

4

1.1 Краткая историческая информация о стеганографии

Стеганография – это, в общем случае, искусство передачи скрытого сообщения. Принято считать, что впервые этот термин использовал Иоганн Тритемий (1462–1516) в своей работе под тем же названием: «Стеганография». Слово «стеганография» в переводе с греческого буквально означает «тайнопись» (от греч. στεγανός (steganos) – скрытый + γράφω (grafo) – пишу). Местом зарождения стеганографии многие называют Египет, хотя первыми «стеганографическими сообщениями» можно назвать и наскальные рисунки древних людей. Первое упоминание о стеганографических методах в литературе приписывается Геродоту, который описал случай передачи сообщения Демартом. Он соскабливал воск с дощечек, писал письмо прямо на дереве, а потом заново покрывал дощечки воском. Другой эпизод, который относят к тем же временам, – передача послания на голове раба. Для передачи тайного сообщения голову раба обривали, наносили на кожу татуировку и, когда волосы отрастали, отправляли с посланием. В Китае письма писали на полосках щелка. Для сокрытия сообщений полоски с текстом письма сворачивались в шарики, покрывались воском и затем глотались посыльными [1].

Темное средневековье породило не только инквизицию: усиление слежки привело к развитию как криптографии, так и стеганографии. Именно в средние века впервые было применено совместное использование шифров и стеганографических методов. XVII–XVIII века известны как эра «черных кабинетов» – специальных государственных органов по перехвату, перлюстрации и дешифрованию переписки. В штат «черных кабинетов», помимо криптографов и дешифровальщиков, входили и другие специалисты, в том числе и химики. Наличие специалистов-химиков было необходимо из-за активного использования так называемых невидимых чернил. Примером может служить любопытный исторический эпизод: восставшими дворянами в Бордо был арестован францисканский монах Берто, являвшийся агентом кардинала Мазарини. Восставшие разрешили Берто написать письмо знакомому священнику в город Блэй. Однако в конце этого письма религиозного содержания монах сделал приписку, на которую никто не обратил внимание: «Посылаю Вам глазную мазь: натрите ею глаза, и Вы будете лучше видеть». Так он сумел переслать не только скрытое сообщение, но и указал способ его обнаружения. В результате монах Берто был спасен. Еще древние римляне писали между строк невидимыми чернилами, в качестве которых использовались фруктовые соки, молоко и некоторые другие натуральные вещества.

Жигаровская С.А.

Их опыт не был забыт. Стеганографические методы активно использовались в США в годы гражданской войны между южанами и северянами. Так, в 1779 году два агента северян Сэмюэль Вудхулл и Роберт Тоунсенд передавали информацию Джорджу Вашингтону, используя специальные чернила. Различные симпатические чернила использовали и русские революционеры в начале XX века. Симпатические чернила или с успехом выполняющее их роль обычное молоко – один из самых распространенных стеганографических методов. Многие книги Ленина были написаны в местах заключения молоком между строк. Чернильницей Владимиру Ильичу служил хлебный мякиш – при малейшем подозрительном звуке В. И. Ленин съедал свои приспособления. Позднее исписанные молоком листы передавались на волю, а там нагревались над лампой и переписывались товарищами по партии. Впрочем, царская охрана тоже знала об этом методе (в ее архиве хранится документ, в котором описан способ использования симпатических чернил и приведен текст перехваченного тайного сообщения революционеров) [1].

К середине XX века стеганография достигла значительных успехов, чему не мало поспособствовали Первая и Вторая мировые войны. Особенных успехов добились немцы, которые во время Второй мировой войны широко применяли «микроточки», представлявшие из себя микрофотографии размером с обычную типографскую точку. При увеличении «микроточка» давала четкое изображение печатной страницы стандартного размера. Такая точка или несколько точек вклеивались в обыкновенное письмо и, помимо сложности обнаружения, обладали способностью передавать большие объемы информации, включая чертежи и рисунки. Сам метод был придуман намного раньше. Микроточки появились сразу же после изобретения Луи Дагером фотографического процесса и впервые в военном деле были использованы во времена франко-прусской войны (в 1870 году), но широкого применения до Второй мировой войны этот метод не имел. Но во время Второй мировой войны данный метод претерпел второе рождение и успех его был весьма заметным. Американцы, впечатленные достижениями своего противника в стеганографии, после войны запретили даже такие относительно невинные операции, как пересылку посредством почты записей шахматных партий, инструкций по вязанию и даже детских рисунков как наиболее простых с точки зрения стеганографа объектов для встраивания шпионских сообщений. Таким образом, стеганография – это наука о способах передачи (хранения) сокрытой информации, при которых скрытый канал организуется на базе и внутри открытого канала с применением особенностей восприятия информации, причем для этой цели могут использоваться следующие приемы: – полное сокрытие факта существования скрытого канала связи; – создание трудностей для обнаружения, извлечения или модификации передаваемых сокрытых сообщений внутри открытых сообщений-контейнеров; – маскировка сокрытой информации в протоколе. Компьютерная стеганография изучает способы сокрытия информации в компьютерных данных, представляющих собой различные файлы, программы, пакеты протоколов и т. п. [1].

1.2 Литературный обзор по теме

На сегодняшний день известны следующие направления стеганографии:

* Классическая стеганография — включает в себя все «некомпьютерные методы»;
* Компьютерная стеганография — направление классической стеганографии, основанное на особенностях компьютерной платформы и использования специальных свойств компьютерных форматов данных;
* Цифровая стеганография — направление классической стеганографии, основанное на внедрении дополнительной информации в цифровые объекты, вызывая при этом некоторые искажения этих объектов. Используется избыточность аудио-, визуальной, а также текстовой информации [2].

1.2.1 Классическая стеганография

Симпатические (невидимые) чернила – чернила, записи которыми являются изначально невидимыми и становятся видимыми только при определённых условиях (нагрев, освещение, химический проявитель).

Процесс записи можно произвести следующим образом: первый слой – наносится важная запись невидимыми чернилами, второй слой – ничего не значащая запись видимыми чернилами [2].

Копирка**.** По внешнему виду эта «копирка» представляет собой лист белой бумаги. На самом же деле «копирка» пропитана специальными химикалиями. Поверх «копирки» кладут лист обычной бумаги. Сообщение пишется карандашом на верхнем листе с соответствующим давлением, передающимся на нижний лист — «копирку». Вместе с тем давление не должно быть слишком сильным, чтобы на «копирке» не остался видимый («вдавленный») оттиск текста. После этого карандашный оригинал уничтожается, а поверх «копирки» обычными чернилами наносится любой текст. Для извлечения послания применяется специальный реактив, нанося его поверхность листа, после чего изображение проявляется на несколько минут. За это время текст нужно сфотографировать или запомнить, так как бумага начинает чернеть, а текст на ней исчезает насовсем и восстановлению больше не подлежит, а затем бумага становится совершенно черной.

Молоко на бумаге. Довольно широко известен метод написания молоком на бумаге. Проявление достигается нагревом над пламенем. Под воздействием высокой температуры сухое вещество молока обугливается быстрее бумаги, выявляя текст. Такой способ «опережающего обугливания» можно применять в отношении многих органических «чернил» и не обязательно на бумаге. Молоко не должно быть с высоким процентом жирности, так как (даже тщательно высушенное) под определенным углом к свету текст может бликовать, выдавая наличие осажденной информации [3].

Микроточки (микродоты). При микрофильмировании изображение можно уменьшить в десятки (и более) раз, что позволяет большой объем информации «спрессовать» в довольно маленький микрофильм. Вариантом фотосъемки, исключающим случайное обнаружение на пленке отснятого материала, является использование технологии микроточек, позволяющее уменьшить изображение в сотни раз [3].

Микроточка представляет собой часть стандартной пленки, которой наряду с малозначительными кадрами имеется миниатюрное изображение, неразличимое невооруженным глазом и извлекаемое с помощью некоторой увеличительной аппаратуры.

Микроточка должна располагаться на заранее оговоренном номере кадра и быть в определенном его месте.

Микроточка также может быть просто вырезана с пленки и наклеена, например, в тексте какого-либо письма или книги вместо обычной точки над буквой или точки в конце любого предложения [3].

1.2.2 Компьютерная стеганография

Использование регистра букв. Данный метод будет продемонстрирован на примере осаждения буквы «А» в текст «information»

Для этого берем двоичное представление кода символа «А» - «01000001». Пускай для обозначения бита, содержащего единицу, используется символ нижнего регистра, а для нуля – верхнего [4].

Поэтому после накладывания маски «01000001» на текст «information», результат будет «iNformaTion». Исспользуя такую технику можно спрятать в текст длинной N, сообщение из N/8 символов. [5].

Использование пробелов. Пробел обозначен символом с кодом 32, но в тексте его можно заменить также символом, имеющим код 255. Пускай для обозначения бита, содержащего единицу, используется символ пробела с кодом 32, а для нуля - 255. В данном случае осаждением информации будет замена пробела на соответствующий символ внедряемой бинарной последовательности.

Использование специфики файловых систем. При хранении файл всегда занимает целое число блоков — это сделано из соображений удобства адресации. Из-за этого хранение небольших файлов реализуется крайне нерационально - в системе Fat32 файл размером 1 байт занимает на диске 4 Кб. При этом очевидно, что все 4 Кб, за вычетом 1 байта никак не используются. Такой метод широко используется для долговременного скрытого хранения небольших объемов информации - однако он легко обнаруживается и крайне небезопасен. Также существует возможность записи данных просто на неиспользуемые области накопителей (например, нулевую дорожку) [5].

1.2.3 Цифровая стеганография

Одним из самых распространённых методов является LSB (Least Significant Bit, наименьший значащий бит) алгоритм, который заменяет наименьший значащий бит в нескольких байтах файла-носителя, чтобы скрыть последовательность байтов, содержащих скрытые данные. Это, как правило, эффективно тогда, когда замена младшего бита не влечет за собой значительное ухудшение качества. Например, в формате BMP изображение хранится как матрица значений оттенков цвета для каждой точки хранимого изображения. Если каждая из компонент пространства RGB хранится в одном байте, она может принимать значения от 0 до 255 включительно, что соответствует 24-х битной глубине цвета. Особенность зрения человека заключается в том, что оно слабо различает незначительные колебания цвета. Для 24-х битного цвета изменение в каждом из трех каналов одного наименее значимого бита (то есть крайнего правого) приводит к изменению менее чем на 1% интенсивности данной точки, что позволяет изменять их незаметно для глаза по своему усмотрению [6].

Допустим, имеется 8-битное изображение в градациях серого. 00h (00000000b) обозначает чёрный цвет, FFh (11111111b) — белый. Всего имеется 256 градаций (2^8). Также предположим, что сообщение состоит из 1 байта — например, 01101011b. При использовании 2 младших бит в описаниях пикселей, потребуется 4 пикселя. Допустим, они чёрного цвета. Тогда пиксели, содержащие скрытое сообщение, будут выглядеть следующим образом: 00000001 00000010 00000010 00000011. Тогда цвет пикселей изменится: первого — на 1/255, второго и третьего — на 2/255 и четвёртого — на 3/255. Такие градации, незаметны для человека, а также могут не отобразиться при использовании низкокачественных устройств вывода [6].

Методы LSB являются неустойчивыми ко всем видам атак и могут быть использованы только при отсутствии шума в канале передачи данных.

Обнаружение LSB-кодированного стего осуществляется по аномальным характеристикам распределения значений диапазона младших битов отсчётов цифрового сигнала.

Если отбросить незначительную относительно размера изображения, служебную информацию в начале файла, то при помощи данного метода можно передать сообщение размером в 1/8 размера контейнера или же размером в 1/4 контейнера (соответственно при использовании 2 последних битов в байтах.

При использовании этого метода данные заключаются в оригинальный звуковой сигнал посредством ввода эха. Данные скрываются при помощи изменения трех параметров эха: начальной амплитудой, степени затухания и задержки. Когда задержка между оригинальным сигналом и эхом уменьшается, сигналы смешиваются. В некоторой точке человеческое ухо уже не может различить эти два сигнала, эхо ощущается как добавочный резонанс. Эту точку достаточно сложно определить, она зависит о качества оригинальной записи, типа записи и слушателя.

При кодировании используется две временные задержки, одна для обозначения логической единицы, другая для логического нуля. Обе задержки должны быть меньше порога чувствительности человеческого уха, при котором он может обнаружить эхо. Вдобавок к уменьшению временной задержки необходимо убедиться, что информацию нельзя раскрыть установлением начальной амплитуды и степени затухания ниже порога слышимости человеческого уха.

Фазовое кодирование – применяется в цифровой аудиостеганографии. Происходит замена исходного звукового элемента на относительную фазу, которая и является секретным сообщением. Фаза подряд идущих элементов должна быть добавлена таким образом, чтобы сохранить относительную фазу между исходными элементами.

Метод встраивания сообщения заключается в том, что специальная случайная последовательность встраивается в контейнер, затем, используя согласованный фильтр, данная последовательность детектируется. Данный метод позволяет встраивать большое количество сообщений в контейнер, и они не будут создавать помехи друг другу. Метод заимствован из широкополосной связи.

Когда происходит резкое изменение фазового соотношения между каждой частотной составляющей, шумы становятся заметными. Тем не менее, если фазу модифицировать не сильно, то человеческое ухо не распознает каких-либо изменений.

Сетевая стеганография. Сетевая стеганография – вид стеганографии, в котором в качестве носителей секретных данных используются сетевые протоколы эталонной модели OSI. В общем виде сетевая стеганография является семейством методов по модификации данных в заголовках сетевых протоколах и в полях полезной нагрузки пакетов, изменению структуры передачи пакетов и гибридных методов в том или ином сетевом протоколе (иногда и нескольких сразу).

Передача скрытых данных в сетевой стеганографии осуществляется через скрытые каналы. Термин «скрытый канал» ввёл Simmons в 1983 году, который определил, что проблема утечки информации не ограничивается использованием программного обеспечения. Скрытый канал может существовать в любом открытом канале, в котором существует некоторая избыточность.

Скрываемые данные называются стеганограммой. Они располагаются в определенном носителе (carrier). В сетевой стеганографии роль носителя выполняет передаваемый по сети пакет.

Основные параметры сетевой стеганографии –это пропускная способность скрытого канала, вероятность обнаружения и стеганографическая стоимость. Пропускная способность - объём секретных данных, который может быть отправлен в единицу времени. Вероятность обнаружения определяется по возможности обнаружения стеганограммы в определенном носителе. Наиболее популярный способ обнаружить стеганограмму – это анализ статистических свойств, полученных данных и сравнение их с типовыми значениями для этого носителя [7].

Методы сетевой стеганографии можно разделить на три группы:

* методы стеганографии, суть которых в изменении данных в полях заголовков сетевых протоколах и в полях полезной нагрузки пакетов;
* методы стеганографии, в которых изменяется структура передачи пакетов;
* смешанные (гибридные) методы стеганографии – при их применении изменяются содержимое пакетов, сроки доставки пакетов и порядок их передачи.

Цифровые водяные знаки. Наиболее востребованный для защиты авторских прав на фотоизображение метод стеганографии – это встраивание так называемых цифровых водяных знаков (ЦВЗ), настроенный на внедрение в мультимедийный файл скрытых маркеров, устойчивых к различным атакам. Встроенные ЦВЗ анализируются специальным декодером, выносящим решение об их корректности. В качестве ЦВЗ могут использоваться данные автора или аутентичный код, а также какая-нибудь управляющая информация [7].

До встраивания информации в изображение в качестве ЦВЗ необходимо преобразовать её к подходящему виду, то есть в двумерный массив бит. Также обычно применяют помехоустойчивое кодирование или широкополосные сигналы для повышения устойчивости ЦВЗ к искажению. Важнейшая предварительная обработка скрытого сообщения и контейнера – это вычисление их обобщенного преобразования Фурье, что позволяет реализовать встраивание ЦВЗ в спектральной области для повышения его устойчивости к искажениям. При предварительной обработке сообщений часто применяют ключи с целью повышения секретности встраивания информации. Следующий этап – это уже непосредственно внедрение ЦВЗ в изображение, часто с использованием так называемого стегоключа (псевдослучайной последовательности бит, полученной от определенного генератора). Незаметность ЦВЗ возможна благодаря большой психовизуальной избыточности изображений для системы восприятия человека [7].

Наиболее распространённый метод встраивания начинается с выбора стегопутей, то есть бит контейнера, которые можно модифицировать без заметных искажений, после чего с помощью ключа из них выбираются биты, заменяемые битами сообщения. Обычно применяют такие методы, как инверсия, вставка или удаление бита, а также использование пороговых бит или различных таблиц значений.

Выделение скрытого в защищаемом изображении ЦВЗ происходит в стегодетекторе с помощью стегоключа, использованного при внедрении информации. Этот ключ может быть как общедоступным, так и предназначенным для узкого круга лиц. Если он общедоступен, то такой стегосистеме сложно противостоять возможным атакам со стороны злоумышленников. Кроме того, само защищаемое изображение может быть изменённым, например, из-за атак на него или операций обработки сигнала. Результатом таких модификаций и атак становится невозможность обнаружения ЦВЗ в стегодетекторе, поэтому учёт этих факторов позволяет строить более эффективные защитные стегосистемы [7].

Стегодетекторы делятся на две категории. Одни из них предназначены для обнаружения наличия ЦВЗ, другие − для его выделения стегодекодерами). В свою очередь, ЦВЗ-стегосистемы делятся на три класса: открытые, полузакрытые и закрытые (имеющие наибольшую устойчивость к внешним воздействиям). Соответственно, для защиты изображений от незаконного использования лучше всего использовать закрытые стегосистемы.

ЦВЗ бывают трёх типов: хрупкие, полухрупкие и робастные. Хрупкие ЦВЗ обычно разрушаются при изменении стегоконтейнера и применяются для аутентификации сигналов. Робастные ЦВЗ устойчивы к различным видам воздействия на контейнер, поэтому именно этому типу меток посвящено множество разработок. Полухрупкие ЦВЗ обычно бывают стойкими к одному виду воздействий, но неустойчивыми по отношению к другим. Строго говоря, робастные ЦВЗ тоже можно отнести к полухрупким, поскольку абсолютной устойчивости меток добиться очень сложно. Для защиты цифровых фотографий лучше использовать робастные ЦВЗ, поскольку у изображений могут быть отредактированы такие параметры, как цветовая гамма, яркость, оно может быть отмасштабировано или повёрнуто. Кроме того, такие ЦВЗ должны обнаруживаться только одной стороной, чтобы злоумышленник не мог их выделить и уничтожить [7].

1.3 Атаки на стегосистемы

Под атакой на стегосистему понимается попытка обнаружить, извлечь, изменить скрытое сообщение. Способность стеганографической системы противостоять атакам называется стеганографической стойкостью. Наиболее простая атака – субъективная. При субъективной атаке контейнер анализируется человеком без применения вспомогательных средств.

Стегосистема считается взломанной, даже если нарушителю удалось, просто, доказать наличие скрытого сообщения в анализируемом контейнере. Предполагается, что нарушитель способен осуществлять любые типы атак и имеет неограниченные вычислительные возможности. Если ему не удается подтвердить гипотезу о том, что в контейнере скрыто секретное сообщение, то стеганографическая система считается устойчивой [8].

В большинстве случаев выделяют несколько этапов взлома стеганографической системы:

* обнаружение факта присутствия скрытой информации;
* извлечение скрытого сообщения;
* видоизменение (модификация) скрытой информации;
* запрет на выполнение любой пересылки информации, в том числе скрытой.

За время существования стеганографии появился ряд методов, позволяющих производить анализ системы с целью взлома.

Атака на основе выбранного пустого контейнера. В этом случае злоумышленник предоставляет собственный контейнер, человеку, который производит внедрение. Например, предложенный контейнер может иметь большие однородные области и тогда будет трудно обеспечить секретность внедрения [8].

Атака на основе известной математической модели контейнера или его части. Атакующий пытается определить отличие сообщения от известной модели. Если биты внутри отсчета изображения коррелированы, тогда отсутствие такой корреляции может служить сигналом об имеющемся скрытом сообщении. Задача внедряющего сообщение заключается в том, чтобы не нарушить статистики контейнера [8].

Адаптивная атака на основе выбранного скрытого сообщения. В данном случае злоумышленник имеет возможность выбирать сообщения для внедрения адаптивно, в зависимости от результатов анализа предыдущих контейнеров.

Атака на основе известного встроенного сообщения. Данный тип атаки характерен для систем защиты интеллектуальной собственности, когда в качестве водяного знака используется известный логотип фирмы. Задачей анализа является получение ключа. Если соответствующий скрытому сообщению заполненный контейнер неизвестен, то решение задачи становиться затруднительным [8].

Атака на основе известного заполненного контейнера. В этом случае у нарушителя есть один или несколько стеганографических контейнера. Предполагается, что встраивание скрытой информации осуществлялось отправителем одним и тем же способом. Задача злоумышленника состоит в обнаружении наличия стегоканала, а также в его извлечении или определения ключа. Зная ключ, нарушитель получит возможность анализа других стегосообщений [8].

Рассмотренные атаки имеют одну особенность: они не изменяют посылаемые стегосообщения, а также не направлены на противодействие работы декодера получателя. В этом заключается их положительная сторона: действия нарушителя вряд ли способны насторожить отправителя и получателя.

Рассматриваемая классификация атак не является единственно возможной.

Сравнение робастности стегосистем производится обычно по отношению к некоторым стандартным тестам. В качестве одного из них является атака, основанная на применении алгоритма сжатия JPEG. Гораздо большее представление о достоинствах того или иного стегоалгоритма можно получить, комплексно используя различные атаки. Общедоступная в Интернете программа Stirmark позволяет более полно анализировать робастность стегоалгоритмов. По утверждению создателей программы на сегодняшний день не существует общеизвестного стегоалгоритма, устойчивого к их комплексным атакам.

1.4 Обзор существующих программных продуктов

Существует довольно большое количество программных продуктов, реализующих описанные выше методы внедрения тайной информации в цифровые объекты:

Одной из утилит для сокрытия информации в графических файлах с использованием криптографии является ImageSpyer. Поддерживается около 30 алгоритмов шифрования и 25 хеш-функций для шифрования контейнера. Скрывает объем, равный числу пикселей изображения. Опционально доступна компрессия скрываемых данных. Утилита совместима с Windows 8. В качестве исходных графических файлов могут использоваться форматы ВМР, JPEG, WMF, EMF, TIFF. Благодаря применению мощного алгоритма упаковки, доступная емкость значительно увеличена. Помимо мощной криптографической защиты безопасность обеспечивается целым рядом параметров секретности, не зная которые злоумышленник не сможет определить наличие скрытой информации. Интерфейс программного продукта ImageSpyer расположен на рисунке 1.1.

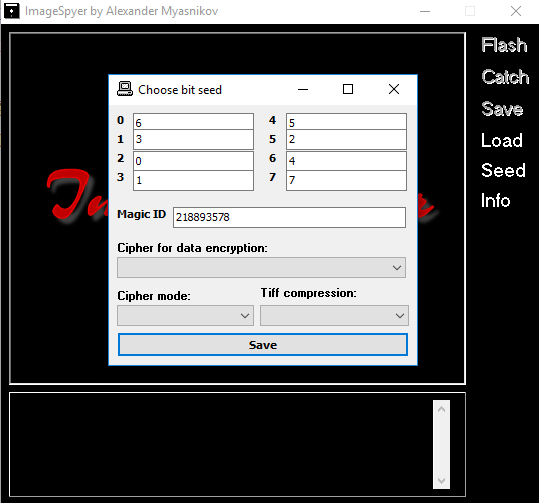


Рисунок 1.1 — Интерфейс приложения ImageSpyer

Стеганографический архиваторный плагин StegoTC G2 TC для Total Comander позволяет скрывать данные в изображении, при этом поддерживаются форматы ВМР, TIFF и PNG [9].

Стеганографическое решение высокой производительности для сокрытия любых данных в JPEG изображении RedJPEG XT, с помощью авторского стеганографического метода. Применяется механизм внедрения непосредственно в само изображение, используется открытые алгоритмы шифрования и мощная LZMA компрессия. Присутствует расширенный режим с дополнительной маскировкой факта внедрения и усиленной процедурой инициализации поточного шифра на основе характеристик изображения. Поддерживает режим случайного секретного распределения [9].



Рисунок 1.2 — Интерфейс приложения RedJPEG XT

Одна из программ для встраивания сообщений в изображения формата JPEG – JSTEG. Алгоритм работы представляет собой замену наименее значимых бит (метод LSB). Программа устойчива к визуальным атакам и имеет высокую пропускную способность, скрытое сообщение может занимать до 12% всего изображения. К недостаткам данного программного продукта относится неустойчивость к анализу гистограмм, а также поддержание только формата JPEG [9].

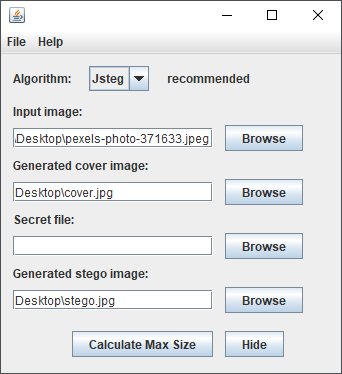


Рисунок 1.3 — Интерфейс приложения JSTEG

Программа позволяющая скрывать любые файлы в GIF, BMP и WAV – S-Tools.. Осуществляет регулируемое сжатие (архивирование) данных, кроме того, производит шифрование с использованием алгоритмов MCD, DES, тройной-DES, IDEA. При шифровании использует пользовательский пароль. S-Tools не нужно устанавливать. Размер файла, который необходимо спрятать в другой файл, должен быть значительно меньше, чем размер файла-контейнера. Чтобы извлечь файл из контейнера, нужна программа S-Tools, а также пароль, использованный при его упаковке в контейнер [9].

Компьютерная программа шифрования, для 32- и 64-разрядных операционных систем семейств Microsoft Windows NT 5 и новее (GUI-интерфейс), Linux и Mac OS X – TrueCrypt. Позволяет шифровать логические диски, разделы жесткого диска или flash-накопителя. Результат работы программы представляется файлом, содержащим всю зашифрованную информацию (каталоги, папки и т.д.). Поддерживает 11 алгоритмов шифрования данных в защищенном паролем томе с возможностью хранения секретных данные в файле (контейнере) или на целом разделе диска, а также на flash-карте, дискете и других съемных устройствах хранения данных. True Type предлагает мощнейшую защиту зашифрованных данных: система двойных паролей, стирание следов шифрования, включая следы работы с мышью и нажатий кнопок клавиатуры, обеспечение двух уровней правдоподобного отрицания наличия зашифрованных данных [9].

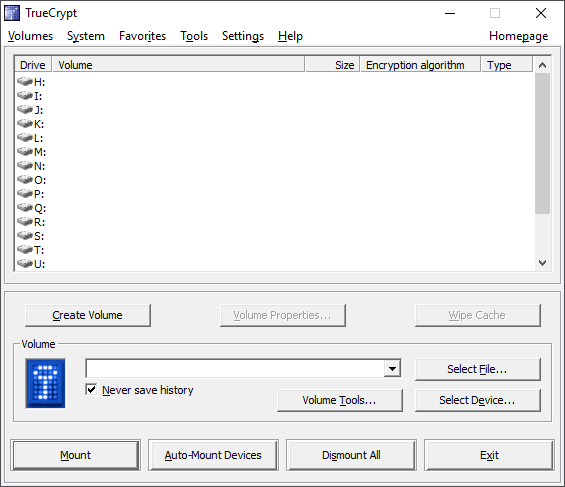


Рисунок 1.4 — Интерфейс приложения TrueCrypt

Стеганографическая программа – Steghide, которая скрывает биты файла данных в некоторых из наименее существенных битов другого файла таким способом, что существование файла данных - не видимо и не может быть доказано. Программа использует blowfish кодирование, MD5 hashing passphrases на blowfish ключи, и псевдослучайное распределение скрытых битов в контейнерных данных [10].

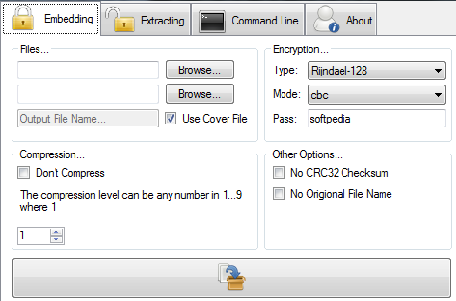


Рисунок 1.5 — Интерфейс приложения Steghide

Бесплатная программа – SecurEngine Professional, позволяющая скрывать различные файлы в графических изображениях или текстовых фалах. Все скрываемые данные предварительно шифруются. Работает на платформах Windows [10].

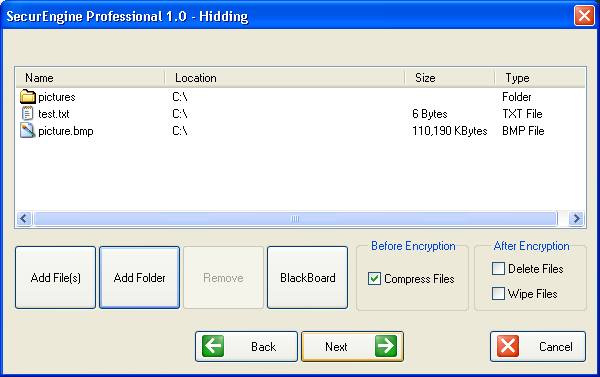


Рисунок 1.6 — Интерфейс приложения SecurEngine Professional

1.5 Патентный поиск по теме проекта

Патентная документация включает описание изобретения к авторскому свидетельству, патенту, заявке или свидетельству на полезную модель.

Для получения информации о существующих разработках в сфере текстовой стеганографии был проведен патентный поиск, результаты которого занесены в таблицу 1.1.

Таблица 1.1 –­ Результаты патентного поиска

|  |  |  |  |
| --- | --- | --- | --- |
| Страна | Номер патента и индекс МПК | | Название патента и описание |
| Китай | CN 102169535  G 06F 21/24 | | МЕТОД ТЕКСТОВОЙ СТЕГАНОГРАФИИ, ОСНОВАННЫЙ НА ЗАМЕНЕ КЛЮЧЕВЫХ СЛОВ.  Изобретение относится к методам текстовой стеганографии, основанным на замене слов, в котором важные слова могут быть заменены ключевыми словами, путем использования различных форм слов. Во время внедрения ключевые слова могут быть выбраны по желанию; ключи генерируются в соответствии с соотношением между ключевыми словами до и после замены; и исходный текст может быть извлечен в соответствии с основным ключом. Заменяемые ключевые слова определяются отправителем, таким образом, чтобы создать как можно трудный для взлома текст. Со стороны аналитика, перехватывающего текст, заметить наличие дополнительной информации крайне затруднительно. После применения данного метода смысл замаскированного текста противоположен смыслу исходного, из-за этого передаваемый текст лишается всякого смысла. |
| США | WO 2010/131939  MY 2009/000167 | | МЕТОД СОКРЫТИЯ СЕКРЕТНОЙ ИНФОРМАЦИИ В ТЕКСТОВОМ КОНТЕЙНЕРЕ.  Тайное сообщение преобразуются в формат кодирования двоичных данных при помощи  только 64 символов ASCII. Затем происходит подсчет пробельных элементов текста, за счет чего определяется минимальный размер исходного контейнера. Сообщение конвертируется в двоичный вид, после чего биты последовательности помещаются в контейнер таким образом, что значение 0 представляется одинарным символом пробела, а 1 обозначают двойным. |
| Продолжение таблицы 1.1 | |  |  |
| Россия | RU 02257010  7H 04L7 H | | СПОСОБ СТЕГАНОГРАФИЧЕСКОГО ПРЕОБРАЗОВАНИЯ БЛОКОВ ДВОИЧНЫХ ДАННЫХ  Изобретение относится к области стеганографического преобразования данных и может быть использовано, когда возникает необходимость скрытого хранения и передачи конфиденциальной информации. Идея изобретения основана на поблоковом преобразовании сообщения, зависящем от секретного ключа, в русскоязычный текст. Технический результат, достигаемый при осуществлении изобретения, заключается в возможности выбора русскоязычных текстов в качестве контейнера для стеганографического преобразования, а также уменьшение зависимости статистических характеристик модифицированного контейнера от скрываемого сообщения, что делает практически невозможным стеганоанализ вероятностно-статистическими методами. |
| Россия | RU 0002463670  H04L 9/00  G09C 1/00 | | СПОСОБ СТЕГАНОГРАФИЧЕСКОЙ ПЕРЕДАЧИ ИНФОРМАЦИИ В СЕТИ ТСР/IP  Изобретение относится к вычислительной технике, а именно к области телекоммуникаций, и предназначено для скрытой передачи секретной информации. Техническим результатом является повышение криптографической и стеганографической стойкости вложений. Способ стеганографической передачи информации, в котором секретный текст с помощью криптографического преобразования превращают  в шифртекст, на основе ключа генерируют псевдослучайную маску, определяющую порядок передачи информационных и маскирующих сегментов. |
|  |

Благодаря патентному поиску проверена степень уникальности изобретений, в области текстовой стеганографии, определена область их применения, включая исследуемую, а также смежные и иные сферы, установлено наличие аналогов.

Кроме того, патентный поиск позволил определить общее состояние исследований в интересуемой области.

Поиск проводился по массиву патентных данных. Эта разновидность информационного поиска – достаточно узкая, поскольку проводится только по патентной документации, которая содержит конкретные данные о техническом решении, достаточно единообразна, обладает необходимой полнотой.

На основании проведенного патентного поиска можно сделать вывод о том, что в сфере текстовой стеганографии было разработано определенное количество методов, использующих для внедрения информации в текстовые контейнеры способы, основанные на лингвистических и синтаксических свойствах исходного контейнера, а также особенностях среды передачи.

1.6 Вывод по разделу

В данном разделе была представлена краткая история стеганографии. Изложены основные направления стеганографии, а также, в частности, методы, используемые для реализации данного подхода к защите информации в разных аспектах.

Рассмотрен факт атаки на стеганографическую систему и ряд способов, позволяющих его осуществить. Дано определение стеганографической стойкости.

Найдены программные продукты, позволяющие создавать контейнеры на основе рассмотренных способов внедрения информации в цифровой объект. При этом можно заметить, что программное обеспечение позволяющее производить стеганографическое преобразование контейнера зависит от операционной системы и ее версии, что существенно затрудняет процесс внедрения информации.

Проведен патентный поиск, направленный на нахождение существующих стеганографических методов, использующих текст в качестве контейнера. На основании проведённого поиска можно сделать вывод о том, что область текстовой стеганографии нуждается в расширении и дополнениями теоретическими сведениями, а также поиске практического применения при решении задач в области тайной передачи и/или хранения информации.

2 Анализ методов текстовой стеганографии

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

22

*ДП 02.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

*Смелов В.В.*

*Анализ методов текстовой стеганографии*

Лит.

Листов

10

БГТУ 84419021, 2018

*419001, 2015*

У

4

2.1 Классификация стегосистем

Также, как и в криптографии, в стеганографии различают системы с секретным ключом и системы с открытым ключом. В стеганографической системе с секретным ключом используется один ключ, который должен быть заранее известен абонентам до начала скрытого обмена секретными сообщениями либо переслан по защищенному каналу. В стегосистеме с открытым ключом для встраивания и извлечения тайного сообщения используются разные ключи, причем вывести один ключ из другого с помощью вычислений невозможно. Один из ключей (открытый) может передаваться свободно по незащищенному каналу связи, а второй, секретный ключ, — по защищенному каналу. Для функционирования безключевых стегосистем не требуется никаких дополнительных данных в виде ключа помимо алгоритма стеганографического преобразования. Безопасность безключевых стегосистем основана на секретности используемых стеганографических преобразований. Как правило для повышения безопасности безключевых систем, перед осаждением информация предварительно выполняется шифруется, что увеличивает защищенность всего процесса передачи информации [11].

Если тайна осаждения информации в контейнер обусловлена лишь алгоритмом или способом реализации такой операции, то можно говорить лишь о простом множестве ключей K, т. е не содержащем подмножеств. Единственным ключом в подобных системах является, собственно, сам алгоритм или способ осаждения/извлечения информации. В некоторых алгоритмах при извлечении скрытой информации дополнительно требуются сведения об исходном контейнере или некоторых других данных. Такие системы представляют ограниченный интерес, поскольку они требуют передачи первоначального вида контейнера, что эквивалентно традиционной задаче ключевого обмена.

Стеганографические системы с открытым ключом не нуждаются в дополнительном канале ключевого обмена. Для их функционирования необходимо иметь два стегоключа: один секретный, который пользователь должен хранить в тайне, а второй — открытый, который хранится в доступном для всех месте. При этом открытый ключ используется в процессе сокрытия информации, а секретный — для ее извлечения [12].

Жигаровская С.А.

2.2 Классификация методов цифровой стеганографии

Большинство методов компьютерной стеганографии базируется на двух принципах. Первый состоит в том, что файлы, которые не требуют абсолютной точности могут быть до определенной степени видоизменены без потери функциональности. Второй принцип основан на отсутствии специального инструментария или неспособности органов чувств человека различать незначительные изменения в таких файлах.

В основе базовых подходов к реализации методов компьютерной стеганографии в рамках той или иной информационной среды лежит выделение малозначимых фрагментов среды и замена существующей в них информации на информацию, которую предполагается защитить. Поскольку в компьютерной стеганографии рассматриваются среды, поддерживаемые средствами вычислительной техники и соответствующими сетями, то вся информационная среда, в конечном итоге, может представляться в цифровом виде.

Таким образом, незначимые для контейнера фрагменты в соответствии с тем или иным алгоритмом заменяются на фрагменты скрываемой информации.

Методы цифровой стеганографии можно разделить на несколько категорий.

По способу отбора контейнера. Существуют методы суррогатной стеганографии, селективной стеганографии и конструирующей стеганографии. В методах суррогатной стеганографии отсутствует возможность выбора контейнера и для сокрытия сообщения выбирается первый попавшийся контейнер, возможно не совсем подходящий для внедрения определенной информации. В этом случае, биты контейнера заменяются битами скрываемого сообщения таким образом, чтобы это изменение не было заметным. В методах селективной стеганографии предполагается, что спрятанное сообщение должно воспроизводить специальные статистические характеристики шума контейнера. Для этого генерируют большое число похожих контейнеров, чтобы затем выбрать наиболее подходящий из них для конкретного сообщения. В методах конструирующей стеганографии контейнер генерируется самой стегосистемой. Шум контейнера моделируется скрываемым сообщением. Это реализуется с помощью процедур, которые не только кодируют скрываемое сообщение под шум, но и сохраняют модель первоначального шума. В предельном случае по модели шума может строиться целое сообщение [13].

По способу доступа к скрываемой информации. Различают методы для потоковых контейнеров и методы для контейнеров с произвольным доступом. Методы, использующие потоковые контейнеры, работают с потоками непрерывных данных (например, интернет-телефония). В этом случае скрываемые биты необходимо в режиме реального времени включать в информационный поток. О потоковом контейнере нельзя предварительно сказать, когда он начнется, когда закончится и насколько продолжительным он будет. Более того, объективно нет возможности узнать заранее, какими будут последующие шумовые биты. Наибольшую проблему при этом составляет синхронизация начала скрытого сообщения. Методы, которые используются для контейнеров с произвольным доступом, предназначены для работы с файлами фиксированной длины (текстовая информация, графические или звуковые файлы). В этом случае заранее известны размеры файла и его содержимое. Скрываемые биты могут быть равномерно выбраны с помощью подходящей псевдослучайной функции. Недостаток таких контейнеров состоит в том, что они обладают намного меньшими размерами, чем потоковые, а также то, что расстояния между скрываемыми битами равномерно распределены между наиболее коротким и наиболее длинным заданными расстояниями, в то время как истинный шум будет иметь экспоненциальное распределение длин интервала. Преимущество подобных контейнеров состоит в том, то они могут быть заранее оценены с точки зрения эффективности выбранного стеганографического преобразования.

По типу организации скрываемой информации. По типу организации информации контейнеры, подобно помехозащищенным кодам, могут быть систематическими и несистематическими.

При скрытии информации используются допущения при расположении и количестве символов в тексте, не учитываемые при прочтении человеком и компьютерном анализе текстового файла. Это может быть дополнительное количество пробелов и знаков табуляции в разных частях строки, чередование некоторых не учитываемых служебных символов, больших и маленьких букв, букв из разных алфавитов, но похоже выглядящих. К методам текстовой стеганографии относят: форматирование, изменение порядка следования маркеров конца строки, метод хвостовых пробелов, метод знаков одинакового начертания и изменения кода пробела. Несмотря на простоту реализации текстовой стеганографии и ее возможное широкое распространение, в настоящее время практически не реализованы методики ее выявления

В систематически организованных контейнерах можно указать конкретные места стеганограммы, где находятся информационные биты самого контейнера, а где – шумовые биты, предназначенные для скрываемой информации (как, например, в широкораспространенном методе наименьшего значащего бита). При несистематической организации контейнера такого разделения сделать нельзя. В этом случае для выделения скрытой информации необходимо обрабатывать содержимое всей стеганограммы [13].

2.3 Разновидности методов текстовой стеганографии

Современные стеганографические средства обычно работают в информационных средах, имеющих большую избыточность. В отличие от информации, которая содержит много шумовых данных (например, звук и изображение), текст содержит малое количество избыточной информации, которую можно использовать для сокрытия данных.

Методы лингвистической стеганографии – сокрытия секретных сообщений в тексте – известны еще со средневековья. В основном такие методы используют либо естественную избыточность языка, либо форматы представления текста. С развитием компьютерных технологий средневековые методы лингвистической стеганографии возродились на качественно новом уровне и позволяют в некоторых случаях скрыть факт тайной переписки.

Методы искажения формата текстового документа. Сокрытие данных путем изменения формата текстовых файлов обычно проводится так, чтобы текстовые редакторы не смогли выявить присутствие дополнительной информации. Для внедрения информации используются методы манипулирования интервалами между словами и предложениями или же пробелами в конце текстовых строк. Использование пробелов для сокрытия данных обусловлено следующими причинами. Во-первых, введение дополнительных пробелов не вносит больших изменений в значение фразы или предложения. Во-вторых, у случайного читателя вряд ли сразу возникнет подозрение относительно вставленных дополнительных пробелов. Сокрытие тайного сообщения (в битовом представлении) можно проводить путем добавления одного или двух символов пробела в конце предложений после символа конца сроки. Один дополнительный пробел кодирует значение бита «0», а два — «1». Этот простой метод имеет недостатки. Во-первых, он не эффективен, т.к. необходим контейнер большого объема. Во-вторых, возможность сокрытия зависит от структуры текста. В-третьих, текстовые редакторы часто автоматически добавляют символы пробела после точки.

Кодировать секретные данные можно дополнительными пробелами в конце каждой строчки текста: два бита кодируются одним пробелом, четыре — двумя, восемь – тремя и т.д. Преимущество такого метода кодирования состоит в том, что оно может быть выполнено с любым текстом; изменения в формате резко не бросаются в глаза читателю, обеспечивается передача большего количества скрытых данных по сравнению с предыдущим методом. Недостаток метода состоит в том, что некоторые программы могут удалять дополнительные пробелы. Помимо этого, скрытые таким образом данные не всегда могут быть восстановлены с печатной копии документа.

Еще один метод сокрытия данных с помощью пробелов манипулирует с текстами, которые выровнены с обеих сторон. В этом методе данные кодируются путем управляемого выбора мест для размещения дополнительных символов пробела. Один символ между словами интерпретируется как 0, а два — как 1. Метод позволяет встраивать несколько бит скрытой информации в каждую строку текста.

Поскольку текст часто выравнивается по ширине листа, не каждый промежуток между словами может использоваться для кодирования скрытых данных. Для того, чтобы определить, в каком из промежутков между словами спрятана информация, а какие промежутки являются частью оригинального текста, используется следующий метод декодирования. Битовая строка, которая извлекается из стеганограммы, разбивается на пары. Пара бит 01 интерпретируется как 1; пара 10 — как 0; а биты 00 и 11 являются пустыми, т.е. такими, которые не несут никакой информации. Например, битовое сообщение 1000101101 сокращается до 001, а строка 110011 — будет пустой [14].

Существуют более тонкие методы сокрытия информации в текстовой среде. В некоторых текстовых редакторах реализованы опции, которые проводят автоматическое форматирование текста в соответствии с определенными критериями. Редактор ТЕХ использует сложный алгоритм вычисления конца строки или страницы, например, количество пробелов, которые необходимо вставить, чтобы сохранить заданный стиль документа. В результате ТЕХ пытается выбрать последовательность мест переносов таким образом, что сумма всех параметров, которые относятся к редактируемому параграфу, была минимальной. Изменяя некоторые значения параметров, можно управлять выбором мест переносов и использовать их для сокрытия данных. До сих пор вопрос о создании безопасной лингвистической стегосистемы остается открытым. Любая обработка текста редактором, его печать или перевод в другой формат.

Однако при использовании данных методов необходимо учитывать особенности таких контейнеров как: HTML, PDF, RTF. Данные форматы могут изменить расположение пробелов и уничтожить скрытый текст. Низкая устойчивость подобных методов к возможным модификациям документа является одной из причин поиска других методов размещения секретной информации в текстовых контейнерах [14].

К синтаксическим методам лингвистической стеганографии относятся методы изменения пунктуации и методы изменения стиля и структуры текста.

В любом языке существуют случаи, когда правила пунктуации являются неоднозначными и имеют слабое влияние на содержание текста. Например, обе формы перечисления “хлеб, масло и молоко” и “хлеб, масло, молоко” являются допустимыми. Можно использовать тот факт, что выбор таких форм является произвольным и использовать альтернативный выбор для кодирования данных в двоичном виде. Например, если появляется форма перечисления с союзом «и», то кодируется 1, иначе – 0. Для сокрытия можно также применять сокращения и аббревиатуры.

В любом языке имеется много возможностей для синтаксического сокрытия данных, но в типовых текстах они встречаются достаточно редко. Средняя скорость передачи данных такими методами равна нескольким битам на килобайт текста.

Хотя многие из правил пунктуации являются неоднозначными и избыточными, их противоречивое использование может стать объектом внимания. Существуют случаи, когда изменение пунктуации может сильно изменить содержание текста. Поэтому такой подход должен использоваться с осторожностью.

К синтаксическим методам относятся методы изменения стиля или структуры текста без существенного изменения его значения или тона.

Семантические методы стеганографии аналогичны синтаксическим методам. Для этих методов элементарными лингвистическими компонентами считаются отдельные слова, поэтому сокрытие данных реализуется путем непосредственной замены слов. Для такой замены необходимы таблицы синонимов. Кодирование секретного сообщения проводится выбором синонима из необходимого места таблицы. Для успешного извлечения внедренной информации потребуется таблица, на основании которой производилось осаждение [14].

2.4 Обоснование пригодности использования XML контейнеров в стеганографии

XML (eXtensible Markup Language, или расширяемый язык разметки) — текстовый формат, предназначенный для хранения структурированных данных, для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML), иногда называемых словарями. Документ представляет собой обычный текстовый файл, в котором при помощи специальных маркеров создаются элементы данных, последовательность и вложенность которых определяет структуру документа и его содержание.

Целью создания XML было обеспечение совместимости при передаче структурированных данных между разными системами обработки информации, особенно при передаче таких данных через Интернет. Словари, основанные на XML (например, RDF, RSS, MathML, XHTML, SVG), сами по себе формально описаны, что позволяет изменять и проверять документы на основе этих словарей, не зная их семантики, то есть, не зная смыслового значения элементов. Важной особенностью XML также является применение так называемых пространств имён.

К основным достоинствам XML можно отнести:

* язык разметки, позволяющий отобразить двоичные данные в текст, читаемый человеком и анализируемый компьютером;
* поддерживает Юникод;
* в формате XML могут быть описаны такие структуры данных как записи, списки и деревья;
* XML имеет строго определённый синтаксис и требования к анализу, что позволяет ему оставаться простым, эффективным и непротиворечивым. Иерархическая структура XML подходит для описания практически любых типов документов, кроме аудио и видео мультимедийных потоков, растровых изображений, сетевых структур данных и двоичных данных;
* XML представляет собой простой текст, свободный от лицензирования и каких-либо ограничений; XML не зависит от платформы;
* не накладывает требований на расположение символов в строке; В отличие от бинарных форматов, XML содержит метаданные об именах, типах и классах описываемых объектов, по которым приложение может обработать документ неизвестной структуры;
* XML имеет реализации парсеров для всех современных языков программирования; XML поддерживается на низком аппаратном, микропрограммном и программном уровнях в современных аппаратных решениях.

К основным недостаткам XML можно отнести:

* избыточность синтаксиса;
* большой размер XML-документа, существенно больше бинарного представления тех же данных. Размер XML-документа существенно больше, чем документа в альтернативных текстовых форматах передачи данных (например, JSON, YAML) и особенно в форматах данных, оптимизированных для конкретного случая использования. При передаче между системами большого количества объектов одного типа (одной структуры), передавать метаданные повторно нет смысла, хотя они содержатся в каждом экземпляре XML описания. Для большого количества задач не нужна вся мощь синтаксиса XML и можно использовать значительно более простые и производительные решения;
* неоднозначность моделирования. Нет общепринятой методологии для моделирования данных в XML, в то время как для реляционной модели и объектно-ориентированной такие средства разработаны и базируются на реляционной алгебре, системном подходе и системном анализе. В результате большой гибкости языка и отсутствия строгих ограничений, одна и та же структура может быть представлена множеством способов (различными разработчиками), например, значение может быть записано как атрибут тега или как тело тега и т. д.;
* не содержит встроенной в язык поддержки типов данных. В нём нет строгой типизации, то есть понятий «целых чисел», «строк», «дат», «булевых значений» и т. д. Иерархическая модель данных, предлагаемая XML, ограничена по сравнению с реляционной моделью и объектно-ориентированными графами, и сетевой моделью данных. Выражение неиерархических данных (например, графов) требует дополнительных усилий [14].

В виду того, что синтаксис XML избыточен – это позволяет рассматривать данный формат в качестве стеганографического контейнера.

Зачастую XML используется скорее в качестве языка разметки, а не формата данных. При описании внешнего вида документа на языке XML, как правило используются атрибуты, что позволяет при помощи определенного алгоритма разместить в файле XML информацию, никак не влияющую на семантику документа.

2.5 Методы текстовой стеганографии, используемые для внедрения информации в документы типа XML

Для скрытия конфиденциальных сообщений в тексте используется или обычная избыточность письменной речи, или же форматы представления текста. Существующие методы текстовой стеганографии, которые получили наибольшее распространение, недостаточно эффективно скрывают сообщения, либо на практике имеют очень узкую применимость и труднореализуемы. Таким образом, актуальной задачей является разработка новых методов, повышающих устойчивость к атакам и имеющих широкую применимость на практике.

Метод изменения порядка следования атрибутов. Метод изменения порядка следования атрибутов основан на безразличии для интерпретатора XML порядка следования атрибутов открывающего тега.

При кодировании битовой последовательности стегосообщения прежде всего имена атрибутов любого открывающего тега документа переставляются местами в алфавитном порядке. В соответствие с этим порядком переставляются и соответствующие им значения. Для кодирования стеганосообщения применяется особенный порядок следования атрибутов. Биту 0 стеганосообщения соответствует алфавитный порядок следования атрибутов, а биту 1 – обратный алфавитный.

Метод фактически не изменяет документа. В то же время внимательный противник сможет заметить различный порядок следования атрибутов в одних и тех же открывающих тегах. Обычно автоматизированные средства создания XML-документов, а также опытные разработчики всегда используют одинаковый порядок следования атрибутов в тэгах. В то же время заметить различный порядок следования атрибутов довольно тяжело. Все зависит от конкретного файла с разметкой.

Таким образом при наличии в теге хотя бы двух атрибутов, возможно осаждение всего лишь одно бита секретного сообщения, что является недостатком данного метода [15].

Метод осаждения пробелов между атрибутами. Метод осаждения пробелов в файлах с разметкой заключается в том, что между именем атрибута и знаком равенства, а также между знаком равенства и значением атрибута можно вставить произвольное количество пробелов (их также может и не быть). Согласно методу бит 0 стеганосообщения отождествляется отсутствием этого пробела, а бит 1 – присутствием пробела.

Открывающий тег с n атрибутами позволяет закодировать 2n бит информации. После внедрения сообщения порядок следования атрибутов изменять нельзя. Данным метод позволяет значительно увеличить емкость стеганографического контейнера по сравнению с предыдущим.

Использование пробелов и табуляции с конце строки. Биты скрываемой информации представляются в виде непечатаемых символов. Такими символами являются «Пробел» и «Горизонтальная табуляция». Таким образом, можно представить биты в виде символов: «1» – «Пробел», «0» – «Горизонтальная табуляция». Каждый байт скрываемой информации преобразуется в последовательность этих символов, где каждому символу соответствует бит скрываемого байта. Далее полученная последовательность помещается в конец строки и становится «невидима». По такому принципу можно скрыть один байт информации в одной строке. Основным недостатком метода является привязанность к количеству строк в документе [16].

2.6 Описание метода текстовой стеганографии на основе замены типа кавычки в XML-документе

Известно, что интерпретатор XML-документа не придает значения какой тип кавычек используется при его создании. Следовательно, если заменить какую-нибудь пару кавычек в валидном XML-документе, например, с двойной на одинарную, то при визуальном анализе документа со стороны пользователя в браузере никакой разницы видно не будет. Используя эту технику, в XML-документ можно осадить бинарную последовательность.

Перед тем, как начинать осаждение информации нужно убедиться в том, что контейнер имеет достаточную емкость. Емкость контейнера определяется как количество пар кавычек во всем документе.

При встраивании последовательности бит условимся, что единице будет соответствовать двойная кавычка, а нулю одинарная. Начиная с первой пары кавычек в документе будем ставить ей в соответствие бит встраиваемого сообщения и, при необходимости, изменять тип кавычки на противоположный. (Например, первая пара кавычек в документе двойная, а первый бит осаждаемой последовательности нулевой, следовательно, необходимо тип кавычек заменить на одинарный).

При использовании данного метода также необходимо заранее определить количество бит, отводимое под один символ сообщения. Установка количества бит позволяет не использовать впустую место в контейнере. Ведь если необходимо передать текст, состоящий только из букв английского алфавита, то для представления одного символа в двоичном виде будет вполне достаточно семи бит, в отличие от русского алфавита, где для представления одного символа необходимо уже как минимум одиннадцать бит.

В конец осажденного сообщения встраивается уникальная последовательность, указывающая на то, что сообщение закончилось.

Эффективность представленного метода зависит от количества атрибутов в XML-документе, позволяя при наличии n атрибутов осадить n бит бинарной последовательности

2.7 Использование формата DOCX в качестве стеганографического контейнера

Формат DOCX представляет собой модернизированную версию формата DOC, причем по сравнению со своим предшественником этот формат гораздо более популярен и доступен. В отличие от файлов DOC формат DOCX не является расширенным файловым форматом. Он представляет собой файл-архив. Формат файла основан на Open XML и использует сжатие по алгоритму ZIP для уменьшения размера файла. Именно это превращает формат в популярное средство переноса данных, т.к. такой файл можно распаковать другими программами (а не только Office Suite).

Этот новый формат файлов свидетельствует о переходе программ и документов пакета Office на международный стандарт Open XML. Формат успешно используется программами Word 2007, 2010 и 2013 для Windows, а также Word 2008 и 2011 для Mac OS X. Кроме того, формат начинают поддерживать и другие производители программного обеспечения.

Исходя из того, что DOCX файл является ZIP архивом с XML документами, можно использовать этот формат для внедрения в него тайной информации.

Для осаждения информации в файл с расширением DOCX можно выполнить данную последовательность шагов:

* создать документ DOCX, например, при помощи текстового редактора Microsoft Office Word 2016;
* разместить в созданном документе текст, применив к нему стили, по необходимости;
* при помощи стандартных средств операционной системы Windows поменять расширение у созданного ранее документа с DOCX на ZIP;
* при помощи стандартных средств операционной системы Windows открыть полученный архив и извлечь XML-документ с именем document.xml из папки word;
* осадить в извлеченный XML-документ тайное сообщение, например, при помощи метода замены кавычек;
* заменить полученным в результате осаждения XML-документ тот, который изначально был в архиве;
* при помощи стандартных средств операционной системы Windows поменять расширение у измененного архива с ZIP на DOCX.

В результате выполнения данной последовательности шагов получен стеганографический контейнер, содержащий некоторую осажденную в него информацию. Поле преобразования, контейнер может быть открыт при помощи Mircoroft Office Word.

2.8 Вывод по разделу

В данном разделе была представлена общая классификация стегосистем, а также методов цифровой и, в частности, текстовой стеганогарафии.

Рассмотрены разновидности существующих методов стеганографии, использующих формат и структуру текстовых документов для внедрения тайной информации.

Проведен анализ формата синтаксиса формата XML. Показана его пригодность для использования в качестве стеганографического контейнера. Рассмотрены уже существующие методы позволяющие внедрять информацию в XML-документы. Предложен новые методы по внедрению информации в XML.

В соответствии с проведенным анализом можно сделать вывод о том, что наиболее значимыми элементами в языке XML для осуществления стойкого, и эффективного с точки зрения внедрения количества информации в документ являются такие элементы языка как: теги, атрибуты, значения атрибутов.

Изучен формат данных DOCX, а также предложен стеганографический метод внедрения информации в документы данного формата на основании того, что сам документ является архивом с набором XML-файлов.

3 Выбор платформы и реализация программного средства

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

32

*ДП 03.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

Смелов В.В.

*Выбор платформы и реализация программного средства*

Лит.

Листов

18

БГТУ 84419021, 2018

*419001, 2015*

У

4

3.1 Технические требования для реализуемого приложения

Разрабатываемое программное средство должно выполнять ряд функций, позволяющих при наличии документа формата DOCX производить стеганографическое осаждение информации на основании метода замены кавычек в файлах языков разметки. К основным функциям, реализующим перечисленный функционал, относятся следующие:

* загрузка документа формата DOCX на сервер;
* получение информации о длине максимально возможного сообщения, которое можно внедрить в загруженный документ;
* выбор языка внедряемого сообщения;
* осаждение тайного сообщения в загруженный документ;
* загрузка стеганографического контейнера с сервера на машину клиента;
* извлечение осажденного сообщения из стеганографического контейнера;

Для возможности использования перечисленных функция необходимо создание пользовательского интерфейса позволяющего при помощи браузера производить внедрение информации в документ-контейнер.

Минимальные системные требования для работы серверной части приложения:

* процессор с частотой 1 ГГц или SoC;
* 1,5 ГБ оперативной памяти;
* 300 МБ места на жестком диске;
* дисплей с разрешением 800 х 600;
* клавиатура и мышь;
* операционные системы: Windows 10 Client (версия 1607+), Windows Server (версия 2008 R2 SP1+), Mac OS X (версия 10.12+), Ubuntu (версии: 17.10, 16.04, 14.04).

3.2 Выбор технологий разработки

Для разработки приложения необходимо выбрать стек технологий позволяющий реализовать вышеописанный функционал. Необходимо учесть особенности формата обрабатываемого документа, в который будет происходить внедрение. Также реализовать клиент-серверную архитектуру для того, чтобы пользователю достаточно было использовать только браузер.

Жигаровская С.А.

На сегодняшний день используются следующие наиболее популярные и эффективные языки программирования для создания web-приложений с установкой дополнительных фреймворков: Ruby, Java, Python, C#. Рассмотрим каждый из них в отдельности.

Ruby – это динамический императивный объектно-ориентированный язык программирования, разработанный Юкихиро Матсумото. Ruby был создан под влиянием таких языков, как Perl, Eiffel и Smalltalk.Характеризуется динамической типизацией и автоматическим управлением памятью. Язык Ruby используется в веб-разработке в составе открытого веб-фреймворка Rails, чаще называемого Ruby on Rails. Ruby on Rails — полноценный, многоуровневый фреймворк для построения веб-приложений, использующих базы данных, который основан на архитектуре Модель-Представление-Контроллер (Model-View-Controller, MVC). Основным преимуществом языка программирования Ruby и фреймворка Ruby on Rails считается скорость разработки. Практика показывает, что скорость разработки проектов на RoR увеличивается на 30–40 процентов по отношению к любому другому языку программирования или фреймворку. В первую очередь прирост скорости разработки определяется обширным набором готовых к работе штатных инструментов RoR, колоссальным набором готовых решений в сообществе, языку Ruby и простоте программирования на нем. Ruby on Rails в его базовой комплектации имеет штатные средства кеширования данных. На старте предоставляются инструменты, позволяющие реализовать кеширование данных на проекте [17].

Однако в данной технологии имеется ряд недостатков. Один из них — это сложность обучения языку. Изучение его выше начального уровня может оказаться непростым, что не позволяет использовать данный инструмент в полной мере при быстром написании проекта. Еще один недостаток — это недостаточность информационных ресурсов посвященных Ruby. При использовании данного языка велика вероятность столкновения с проблемой решение которой может занять значительную часть времени [18].

История создания языка Java начинается в июне 1991 года, когда Джеймс Гослинг создал проект для использования в одном из своих многочисленных сет-топ проектов. Язык, который рос вне офиса Гослинга, как дуб, Oak - первоначальное название Java до 1995 года, после в дальнейшем история Java продолжалась под именем Green, а позже был переименован как Java. Но официальной датой создания языка Java считается 23 мая 1995 года, после выпуска компанией Sun первой реализации Java 1.0. Она гарантировала «Напиши один раз, запускай везде», обеспечивая недорогой стоимостью на популярных платформах. 13 ноября 2006 года, Sun выпустила большую часть как свободное и открытое программное обеспечение в соответствии с условиями GNU General Public License (GPL). После 8 мая 2007 года судьба Java сложилась иначе. Компания завершила процесс, делая все чтобы исходный код был бесплатным и открытым, кроме небольшой части кода, на который компания не имела авторских прав [19].

В Java все является объектом. Дополнение может быть легко расширено, так как он основан на объектной модели. В отличие от многих других языков, включая C и C++, Java, когда был создан, он не компилировался в платформе конкретной машины, а в независимом от платформы байт-коде. Этот байт код распространяется через интернет и интерпретируется в Java Virtual Machine (JVM), на которой он в настоящее время работает. Процессы изучения и введение в язык программирования Java остаются простыми. Методы проверки подлинности основаны на шифровании с открытым ключом. Архитектурно-нейтральный и не имеющий зависимости от реализации аспектов спецификаций — все это делает Java портативным. Компилятор в Java написан на ANSI C с чистой переносимостью, который является подмножеством POSIX [19].

«Многословность и громоздкость» Одна из проблем Java. И причины здесь можно выделить две: жесткая политика Объектно-Ориентированного Подхода, отсутствие функций первого класса (first-class function). Политика ООП введена в Java с целью избавления от некоторых опасностей, возникающих при создании крупного проекта. В Java же невозможно создавать функции, не являющиеся методами какого-либо класса. К примеру, в С++ такого запрета нет [20].

Функции первого порядка сходны с классами первого порядка. Это функции, которые можно передавать как параметры. В С++ это решается при помощи поддержки указателей на функции. В Java все же есть способы передачи ссылки на функцию, но обычно там, где в С++ используется указатель на функцию, в Java принято использовать Интерфейсы. Это более многословная технология, но при этом она избавляет от опасностей при использовании указателей.

Java, несмотря на различные способы оптимизации, всё же довольно ресурсоёмка и медлительна. Причины в следующем:

* автосборка мусора;
* компиляция «на лету» (Just In Time compilation);
* отказ от таких опасных механизмов как: арифметика указателей, неявное преобразование типов с потерей точности, функции первого класса [20].

Python – это язык программирования общего назначения, нацеленный в первую очередь на повышение продуктивности программиста. Порог вхождения низкий, а код во многом лаконичный и понятный даже тому, кто никогда на нём не писал. За счёт простоты кода, дальнейшее сопровождение программ, написанных на Python, становится легче и приятнее по сравнению с Java или C++. А с точки зрения бизнеса это влечёт за собой сокращение расходов и увеличение производительности труда сотрудников. Существует множество доступных сред разработки, сервисов и фреймворков. Не составит труда найти качественный и бесплатный продукт для работы дома, в офисе и в дороге. Наиболее популярным фреймворком для разработки web-приложений на python является Django. Django появился в 2005 году, и постепенно стал одним из лучших фреймворков, который помогает разработчикам выполнять ту или иную работу в течение нескольких минут. К плюсам Django можно отнести следующее:

* фреймворк разработан, чтобы помочь разработчикам создать приложение настолько быстро, на сколько это возможно. Это включает в себя формирование идеи, разработку и выпуск проекта, где Django экономит время и ресурсы на каждом из этих этапов [21];
* django работает с десятками дополнительных функций, которые заметно помогают с аутентификацией пользователя, картами сайта, администрированием содержимого, RSS и многим другим. Данные аспекты помогают осуществить каждый этап веб разработки;
* работая в Django, вы получаете защиту от ошибок, связанных с безопасностью и ставящих под угрозу проект. Такие распространенные ошибки, как инъекции SQL, кросс-сайт подлоги, clickjacking и кросс-сайтовый скриптинг осуществить крайне затруднительно;
* фреймворк Django наилучшим образом подходит для работы с самыми высокими трафиками [21-22].

К минусам фреймворка можно отнести следующее:

* использование шаблона маршрутизации с указанием URL;
* django монолитный;
* все базируется на ORM Django;
* компоненты развертываются совместно;
* необходимо умение владеть всей системой для работы [23].

C#, является объектно-ориентированным языком, но поддерживает также и компонентно-ориентированное программирование. Разработка современных приложений все больше тяготеет к созданию программных компонентов в форме автономных и самоописательных пакетов, реализующих отдельные функциональные возможности. Важная особенность таких компонентов — это модель программирования на основе свойств, методов и событий. Каждый компонент имеет атрибуты, предоставляющие декларативные сведения о компоненте, а также встроенные элементы документации. C# предоставляет языковые конструкции, непосредственно поддерживающие такую концепцию работы. Благодаря этому C# отлично подходит для создания и применения программных компонентов.

Вот лишь несколько функций языка C#, обеспечивающих надежность и устойчивость приложений: сборка мусора автоматически освобождает память, занятую уничтоженными и неиспользуемыми объектами; обработка исключений дает структурированный и расширяемый способ выявлять и обрабатывать ошибки; строгая типизация языка не позволяет обращаться к неинициализированным переменным, выходить за пределы массива или выполнять неконтролируемое приведение типов [24].

Чтобы обеспечить совместимость программ и библиотек C# при дальнейшем развитии, при разработке C# много внимания было уделено управлению версиями. Вопросы управления версиями существенно повлияли на такие аспекты разработки C#, как раздельные модификаторы virtual и override, правила разрешения перегрузки методов и поддержка явного объявления членов интерфейса [25].

При создании web-приложений на языке C# возможно использование платформы ASP.NET Core которая представляет технологию от компании Microsoft, предназначенную для создания различного рода небольших веб-сайтов, а также крупных веб-порталов и веб-сервисов. ASP.NET Core может работать поверх кроссплатформенной среды .NET Core, которая может быть развернута на основных популярных операционных системах: Windows, Mac OS X, Linux. И таким образом, с помощью ASP.NET Core можно создавать кроссплатформенные приложения [26].

Основными преимуществами платформы являются:

* легковесный и модульный конвейер HTTP-запросов;
* возможность развертывать приложение как на IIS, так и в рамках своего собственного процесса;
* использование платформы .NET Core и ее функциональности;
* распространение пакетов платформы через NuGet;
* интегрированная поддержка для создания и использования пакетов NuGet;
* единый стек веб-разработки, сочетающий Web UI и Web API;
* конфигурация для упрощенного использования в облаке;
* расширяемость;
* наличие встроенных библиотек для работы с такими форматами данных как: DOCX, ZIP позволяющих модифицировать данные форматы;
* кроссплатформенность: возможность разработки и развертывания приложений ASP.NET на Windows, Mac и Linux;
* совместимость с UI-фреймворками [26].

Angular. Представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк. Angular 5 предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее. Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript. Поэтому перед началом работы рекомендуется ознакомиться с основами данного языка. Однако фреймворк не ограничен языком TypeScript, также поддерживаются такие языки как Dart или JavaScript. Последняя версия Angular - Angular 5 вышла 1 ноября 2017 года.

К основным преимуществам Angular можно отнести:

* большое комьюнити. В него входят как участники постоянной команды разработки, так и просто те, кто хотят внести свою лепту в развитие фреймворка с открытым исходным кодом. По Angular проходит множество конференций, о нем говорят на хакатонах и спорят в тематических ИТ-коммьюнити. Существует много книг и он-лайн ресурсов по Angular для разработчиков.
* использование директив. В качестве языка шаблонов в Angular используется HTML. Он расширяется с помощью директив, которые добавляют в код сведения о требуемом поведении (например, о необходимости загрузить определенный модуль сразу после загрузки страницы). Директивы позволяют сконцентрироваться на проработке логики и работать более продуктивно. Их можно использовать повторно, что также повышает читабельность кода.
* cхема разработки MVC. В Angular используется схема MVC, разделяющая логику, представление и данные приложения. Это позволяет создавать одностраничные веб-приложения.
* наличие готовых решений. Для Angular существует огромное количество готовых решений, которые позволяют решать довольно разнообразные задачи, используя уже готовые модули.
* двустороннее связывание данных. В Angular применяется двустороннее связывание: любые изменения в пользовательском интерфейсе сразу же отражаются на объектах приложения и наоборот. Фреймворк сам следит за событиями браузера, изменениями модели и действиями пользователя на странице, чтобы сразу обновлять нужные шаблоны. При этом в коде не требуется хранить ссылки на DOM-элементы и явно ими манипулировать [27-29].

Проанализировав особенности вышеперечисленных технологий можно сделать вывод о том, что наиболее подходящей, для реализации программного средства, осуществляющего стеганографическое преобразование контейнера формата DOCX является ASP.NET Core, благодаря наличию широкого набора инструментов позволяющих работать с представленными форматами данных. В качестве фреймворка для реализации клиентского приложения хорошо подойдет Angular.

3.3 Выбор IDE для разработки

Одной из наиболее популярных сред разработки для языка C# является Microsoft Visual Studio. Microsoft Visual Studio 2017 — это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных телеметрии по использованию. Эти инструменты предназначены для максимально эффективной совместной работы; все они доступны в интегрированной среде разработки (IDE) Visual Studio [30].

Последней версией поддерживающей необходимый функционал является Microsoft Visual Studio Community 2017 15.4.5. Семейство инструментов Visual Studio 2017 содержит IDE, сервис для организации совместной работы – Visual Studio Team Services, его on-premise версия, известная, как TFS, комплексное решение для реализации полноценного цикла разработки мобильных приложений – Visual Studio Mobile Center, кроссплатформенный редактор кода Visual Studio Code (доступен для Mac, Linux и Windows), а также превью-версия Visual Studio for Mac. Использоание Visual Studio при разработке проектов на ASP.NET Core дает определенные преимущества, которые касаются структуры проекта, основанной на .csproj, что обеспечивает совместимость с build-системами для .NET, основанными на MSBuild. Дополнительно, формат.csproj значительно упрощает возможности редактирования файлы для объявления зависимостей, target-платформ и свойств проекта. В CLI имеются дополнительные команды и возможность выбора собственных шаблонов проекта [30].

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Некоторые виды приложений создаваемые при помощи среды разработки:

* приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;
* веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
* приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect и "Интернета вещей";
* игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

Текстовый редактор включает много интерактивных функций (если они требуются) и функций повышения производительности, помогающих ускорить написание кода. Функции различаются в зависимости от языка, и необязательно использовать их все (введите "Редактор" на панели быстрого запуска, чтобы включить или отключить функции). Некоторые распространенные возможности повышения производительности приведены ниже.

* рефакторинг включает такие операции, как интеллектуальное переименование переменных, перемещение выделенных строк кода в отдельную функцию, перемещение кода в другие расположения, изменение порядка параметров функции и т. д.;
* IntelliSense — это общий термин для набора очень популярных функций, отображающих сведения о типах в коде непосредственно в редакторе и в некоторых случаях автоматически создающих небольшие отрывки кода. По сути, IntelliSense представляет собой базовую документацию, встроенную в редактор, что избавляет от необходимости поиска информации о типах в отдельном окне справки. Функции IntelliSense зависят от языка. Дополнительные сведения: IntelliSense для Visual C#, Visual C++ IntelliSense, IntelliSense для JavaScript, Возможности IntelliSense в Visual Basic;
* волнистые линии предупреждают об ошибках или потенциальных проблемах в коде в режиме реального времени по мере ввода, что позволяет исправлять их немедленно, не дожидаясь обнаружения ошибок во время компиляции или выполнения. Если навести указатель мыши на волнистую линию, на экран будут выведены дополнительные сведения об ошибке.
* закладки позволяют быстро переходить к определенным строкам в файлах, с которыми вы работаете;
* в контекстном меню текстового редактора можно вызвать окно «Иерархия вызовов» для отображения методов, которые вызываются методом или вызывают метод, в котором установлен курсор;
* codeLens позволяет находить ссылки на код, изменения кода, связанные ошибки, рабочие элементы, проверки кода и модульные тесты — все это, не выходя из редактора;
* окно «Показать определения» позволяет просмотреть определение метода или типа в окне редактора, не покидая текущий контекст.
* пункт контекстного меню «Перейти к определению» позволяет перейти непосредственно к тому месту, где определена функция или объект [31].

Исходя из вышеперечисленных особенностей среды разработки Visual Studio, можно сказать, что она обладает всем необходимым функционалом для реализации разрабатываемого программного средства при использовании выбранных технологий.

3.4 Разработка приложения

Процесс создания приложения состоит из нескольких, связанных друг с другом, последовательных этапов:

* Разработка алгоритма осаждения/извлечения информации в текстовый контейнер формата DOCX;
* разработка архитектуры приложения;
* реализация серверного приложения;
* реализация клиентского приложения.

3.4.1 Разработка алгоритма осаждения/извлечения информации в текстовый контейнер формата DOCX

Для реализации возможности осаждения информации в контейнер формата DOCX приложение должно обладать рядом функций:

* загрузка DOCX-документа на сервер;
* чтение XML-документа из загруженного DOCX-файла;
* вычисление максимального размера информации возможной для осаждения в XML-документ с учетом языка;
* внедрение бинарной последовательности в XML-документ;
* замена XML-документа в архиве формата DOCX.
* отправка DOCX-документа, содержащего тайную информацию клиенту;
* извлечение внедренной информации.

В результате проведения операции осаждения формируется документ с расширением DOCX содержащий внедренную информацию.

Для осуществления процедуры осаждения, в адресной строке браузера необходимо ввести адрес приложения MarkupStego. После загрузки приложения возможно выполнение операций внедрения и извлечения, переключаться между этими двумя функциями можно при помощи меню располагающемся сверху, при помощи кнопок «Внедрение» и «Извлечение». При переходе на вкладку «Внедрение» для проведения внедрения информации необходимо загрузить файл с расширением DOCX. После чего указать язык внедряемого сообщения, ввести само сообщение и ввести название контейнера, содержащего информацию. Для проведения операции извлечения необходимо загрузить контейнер, в который было произведено осаждение предварительно выбрав язык сообщения, после чего внедренная ранее информация отобразиться в окне браузера.

3.4.2 Разработка алгоритма внедрения информации

Алгоритм внедрения тайного сообщения представлен на рисунке 3.1 в виде блок-схемы.

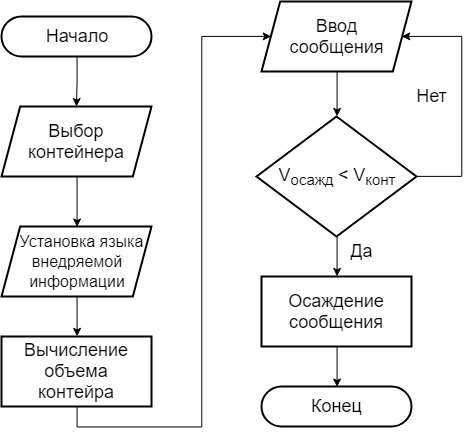


Рисунок 3.1 — Алгоритм внедрения информации в DOCX-контейнер

Для передачи технической информации об осаждении между клиентом и сервером создан класс StegoContainerInfoDTO код которого представлен на рисунке 3.2.

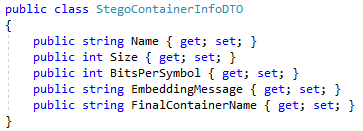


Рисунок 3.2 — Описание класса StegoContainerInfoDTO

Для хранения информации о контейнере во время проведения осаждения создан вспомогательный класс XMLFile, являющийся абстракцией над реальным документам XML, описание которого представлено на рисунке 3.3.

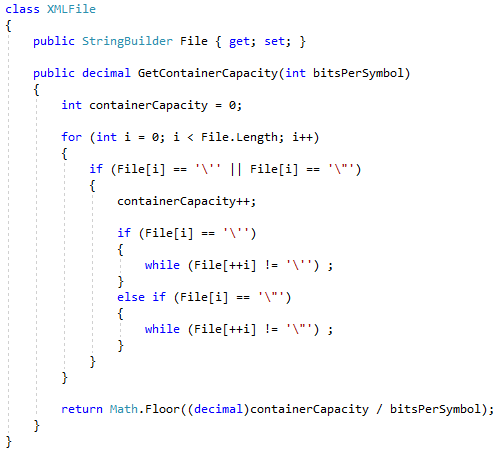


Рисунок 3.3 — Описание класса XMLFile

Для определения размера сообщения которой можно осадить в XML-документ, в классе XMLFile имеется метод, который на основании подсчета количества пар двойных и одинарных кавычек в документе, учитывая язык внедряемого сообщения возвращает максимальное число символов из которых может состоять внедряемое сообщение. Также в своем составе класс XMLFile имеет свойство File содержащее текст считываемого XML-документа. Данное имеет тип StringBuilder позволяющее изменять содержимое свойства, не создавая при этом лишних объектов, что позволяет не тратить впустую память сервера.

Для манипуляции файлами создан класс FileManager содержащий набор методов для перемещения, а также изменения содержимого документов. На рирунке 3.4 показаны методы ReadXMLFile и WriteXMLFile, позволяющие считывать/изменять содержимое реального XML файла и сохранять его состояние в объект класса XMLFile.

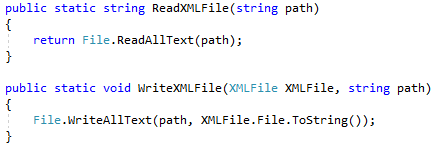


Рисунок 3.4 — Методы позволяющие манипулировать содержимым XML файлов

Для того, чтобы извлечь XML-документ из DOCX-файла необходимо изменить расширение с DOCX на ZIP. Методы, реализующие смену расширения файла представлены на рисунке 3.5.

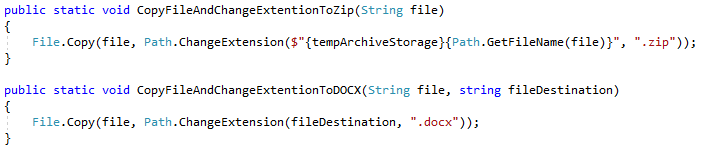


Рисунок 3.5 — Методы, реализующие смену расширения файлов

Для извлечения содержимого XML-документа из ZIP-архива реализован метод ReadDocumentFromZipFile, представленный на рисунке 3.6. Метод принимает на вход путь, по которому можно обратиться к XML-докуметну и используя готовый класс метод Open класса ZipFile из пространства имен System.IO.Compression считывает текст XML-файла.

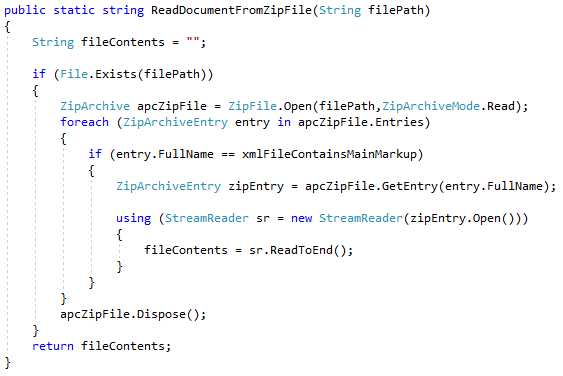


Рисунок 3.6 — Реализация метода ReadDocumentFromZipFile

Метод RemoveDocumentFilefromZipArchive представленный на рисунке 3.7 удаляет XML-документ, содержащий разметку, из архива.

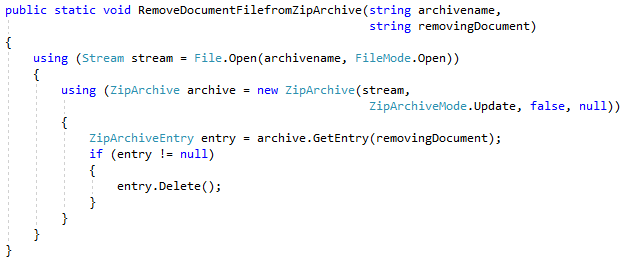


Рисунок 3.7 — Реализация метода RemoveDocumentFilefromZipArchive

После стеганографического преобразования измененный XML-документ помещается в архив при помощи метода AddStegoContainerToArchive реализация которого представлена на рисунке 3.8.

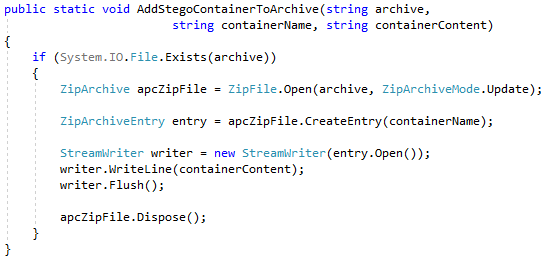


Рисунок 3.8 — Реализация метода AddStegoContainerToArchive

Перед внедрением сообщения в XML-документ, его необходимо конвертировать в бинарный вид, при помощи метода MakeBinaryString представленного на рисунке 3.9.

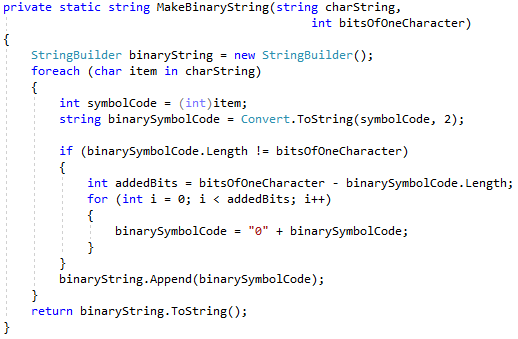


Рисунок 3.9 — Реализация метода MakeBinaryString

Для внедрения сообщения в XML-документ, используется метод EmbedMessage, представленный на рисунке 3.10.

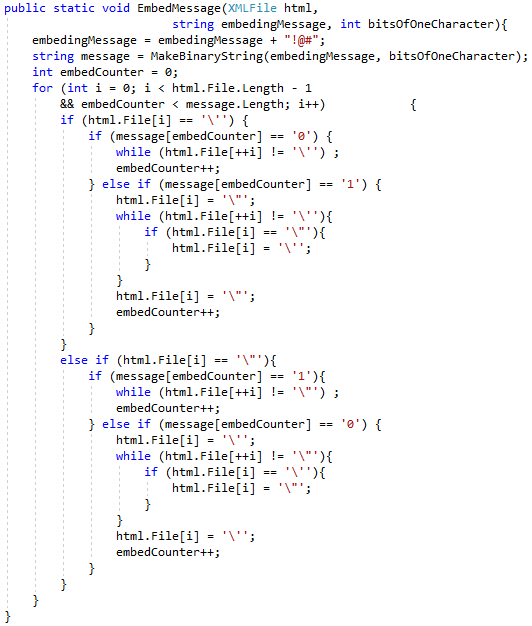


Рисунок 3.10 — Реализация метода MakeBinaryString

Представленный метод заменяет в документе двойные и одинарные кавычки в соответствии с бинарной последовательностью полученной из осаждаемого сообщения. Данный метод модифицирует принимаемый XML-документ, который при помощи метода AddStegoContainerToArchive размещается в ZIP архиве.

3.4.3 Разработка алгоритма извлечения информации

Алгоритм извлечения тайного сообщения из стеганографического контейнера представлен на рисунке 3.11 в виде блок-схемы.

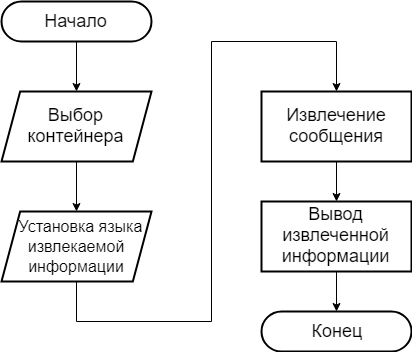


Рисунок 3.11 — Алгоритм извлечения информации в DOCX-контейнер

Для изменения положения файлов можно использовать методы который, были описаны выше. Извлечение XML-документа из архива, в который раннее было произведено осаждение, проводится при помощи метода ReadDocumentFromZipFile. Стеганографическое извлечение бинарной последовательности производиться при помощи метода ExtractMessage представленного на рисунке 3.12. Представленный метод проходится последовательно до передаваемому на вход документу и считывает все кавычки. Во время считывания одинарной кавычке ставиться в соответствие двоичный 0, а двойной двоичная 1. Считываемая бинарная последовательность сохраняется в переменную ExtractMessage.

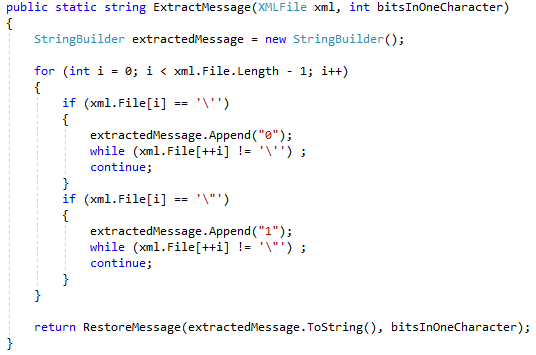


Рисунок 3.12 — Реализация метода ExtractMessage

После окончания считывания сообщения, извлеченная бинарная последовательность передается в метод RestoreMessage представленный на рисунке 3.13 для преобразования информации с символьный вид.

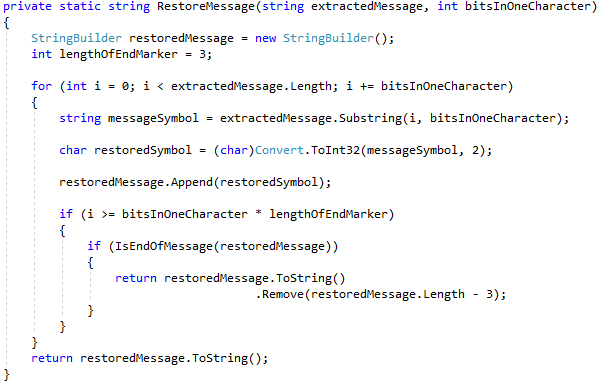


Рисунок 3.13 — Реализация метода RestoreMessage

3.4.4 Разработка клиентской части приложения

Для разработки клиентской части приложения используется фреймворк Angular. При создании приложений с использованием данной технологии, логические части пользовательского интерфейса разделяются на слабо связные компоненты, которые можно эффективно использовать во всем приложении.

Для организации удобного перемещения по компонентам, представляющим разную логику, используется элемент «router-outlet», код которого представлен на рисунке 3.14.



Рисунок 3.14 — Использование тега «router-outlet» на клиентской части

Переключение компонентов осуществляется за счёт использования атрибута «routerLink» в теге «а», который сообщает «router-outlet» какую разметку необходимо отобразить, использование атрибута показано на рисунке 3.15.

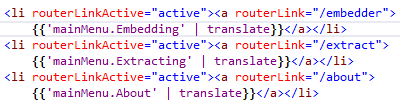


Рисунок 3.15 — Использование атрибута «routerLink»

Настройка «router-outlet» производится в отдельном файле, где указывается какой компонент нужно отображать на определенный путь в «routerLink». Конфигурация «router-outlet» вынесена в отдельный файл «app-routing.module.ts», основная часть которого представлена на рисунке 3.16.

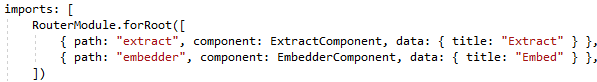


Рисунок 3.16 — Основная часть конфигурации «router-outlet»

После размещения данных тегов и атрибутов в разметке можно увидеть, что на странице появилось меню, которое отвечает за переключение между компонентами внедрения и извлечения сообщения, а также информации о приложении. Данное меню представлен на рисунке 3.17.

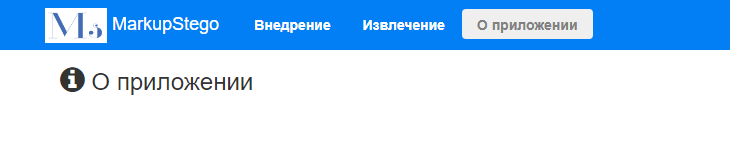


Рисунок 3.17 — Меню переключения компонентов приложения

Основная разметка приложения находится в двух компонентах: «Embedder» и «Extract», которые позволяют пользователю производить внедрение и извлечение сообщения соответственно.

3.4 Вывод по разделу

Для реализации приложения была выбрана платформа ASP.NET Core, которая представляет собой технологию от компании Microsoft, предназначенную для создания различного веб-приложений. ASP.NET Core включает в себя фреймворк MVC, объединяющий функциональность MVC, Web API и Web Pages, что позволяет разделить логику приложения и его визуальное представление. Для написания серверной части используется язык программирования C#, являющийся изящным объектно-ориентированным языком со строгой типизацией, позволяющий создавать различные безопасные и надежные приложения, работающие на платформе .NET.

Фреймворком для реализации клиентской части был выбран Angular. Благодаря компонентам Angular приложение разбивается на независимые части, которые легко использовать в любой его части. Так же одним из преимуществ Angular, является TypeScript. Этот язык даёт возможности современного автозаполнения, навигации и рефакторинга.

В качестве инструментария разработки Visual Studio 2017, которая имеет все необходимые инструменты для написания и рефакторинга кода, а также поддержки и мониторинга жизненного цикла приложения. При разработке в Visual Studio 2017 проекты приложений имеют встроенную поддержку с такими популярными инструментами, как Bower, Grunt, Webpack, который позволяют управлять скриптами JavaScript и стилями CSS, автоматизировать и оптимизировать процесс веб-разработки.

На основании разработанных алгоритмов внедрения/извлечения информации было реализовано WEB-приложение позволяющее создавать стеганографические контейнеры формата DOCX.

4 Тестирование программного средства

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

32

*ДП 04.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

Смелов В.В.

*Тестирование программного средства*

Лит.

Листов

9

БГТУ 84419021, 2018

*419001, 2015*

У

4

4.1 Дымовое тестирование

В области программного обеспечения, дымовое тестирование рассматривается как короткий цикл тестов, выполняемый для подтверждения того, что после сборки кода (нового или исправленного) устанавливаемое приложение, стартует и выполняет основные функции [32].

Практика применяемая, к примеру, в Microsoft и некоторых других компаниях, занимающихся разработкой ПО, заключается в ежедневной сборке (билдовании) программы, которая дополняется дымовым тестированием. Ежедневно после того, как каждый файл собран (сбилдован, построен), связан (слинкован), и объединен в выполнимую программу, сама программа подвергается достаточно простому набору тестов, цель которых заключается в том, чтобы увидеть, «дымит» ли программа во время работы. Эти тесты и называются дымовыми (от англ. smoke — дым).

Один из самых существенных рисков, с которым сталкивается команда разработчиков, заключается в том, что сами разработчики работают с кодом отдельно, независимо друг от друга, в результате чего сложная программа не работает, как ожидается при сборке наработанного кода. В зависимости от того, когда была обнаружена несовместимость в проекте, отладка программы может происходить дольше, чем при более ранней интеграции, особенно в случае изменения интерфейса программы или после имплементации серьезных правок основных частей программы.

Вывод о работоспособности основных функций делается на основании результатов поверхностного тестирования наиболее важных модулей приложения на предмет возможности выполнения требуемых задач и наличия быстронаходимых критических и блокирующих дефектов. В случае отсутствия таковых дефектов дымовое тестирование объявляется пройденным, и приложение передается для проведения полного цикла тестирования, в противном случае, дымовое тестирование объявляется проваленным, и приложение уходит на доработку [32].

Данный тип тестирования является видом тестирования продукта по глубине, а не просто видом тестовых испытаний. Как говорилось выше, данный тип тестирования определяет, пригоден ли продукт для дальнейшего, более полного тестирования. В случае, если он не проходит smoke testing — продукт необходимо отправить на доработку [33].

Жигаровская С.А.

Для успешного прохождения дымового тестирования приложение должно предоставлять возможность пользователю создавать стеганографический контейнер, содержащий внедренное сообщение, а также извлекать осажденную информацию.

В качестве документа, в который будет происходить внедрение информации будет использоваться DOCX-файл содержащий отформатированный текст. Содержимое текстового документа представлено на рисунке 4.1.

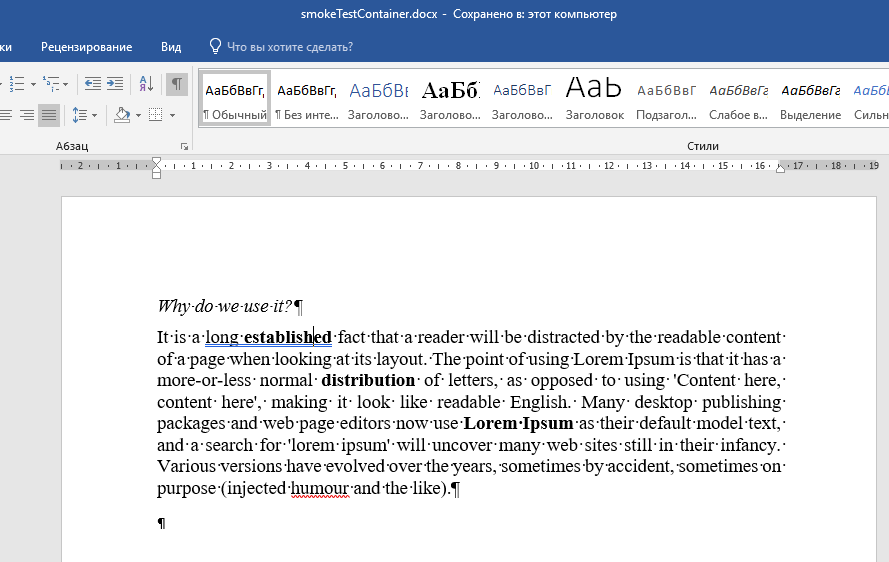


Рисунок 4.1 — Содержимое тестового DOCX-документа

Откроем приложение и дождемся загрузки главной страницы. Загруженная главная страница представлена на рисунке 4.2.

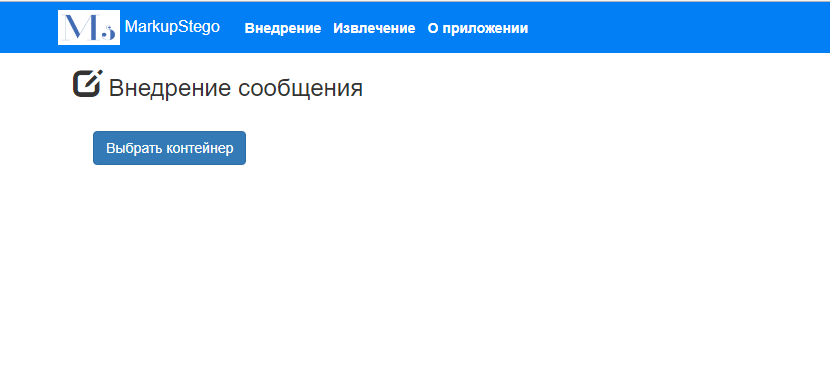


Рисунок 4.2 — Главня страница приложения

Загрузим документ в приложение и увидим его основные стеганографические свойства. Свойства загруженного документа отображены на рисунке 4.3.

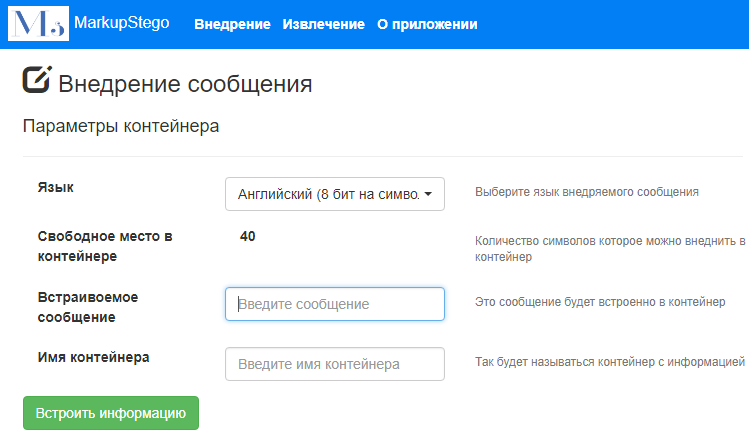


Рисунок 4.3 — Основные стеганографические свойства контейнера

Заполним поля «Встраиваемое сообщение» и «Имя контейнера», и нажмем кнопку «Встроить сообщение». После чего начнется скачивание документа, содержащего внедренное сообщение. Заполненная информация о контейнере расположена на рисунке 4.5

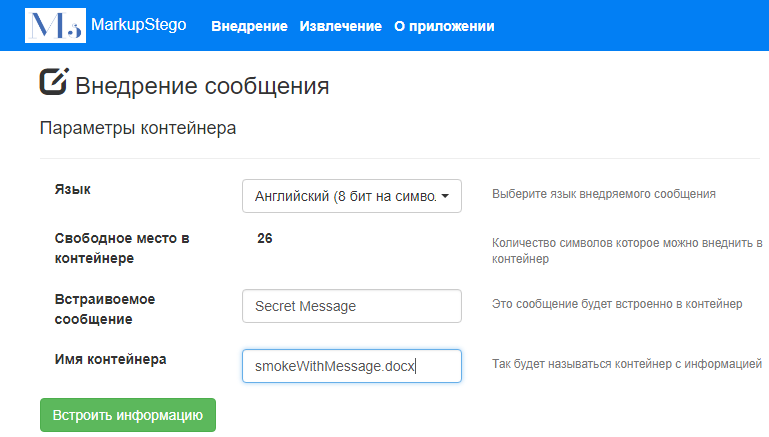


Рисунок 4.5 — Форма с заполненной информации о осаждении

При открытии скачанного документа никакой разницы со стороны пользователя в приложении Microsoft Word до и после осаждения не заметно, что показано на рисунке 4.6.

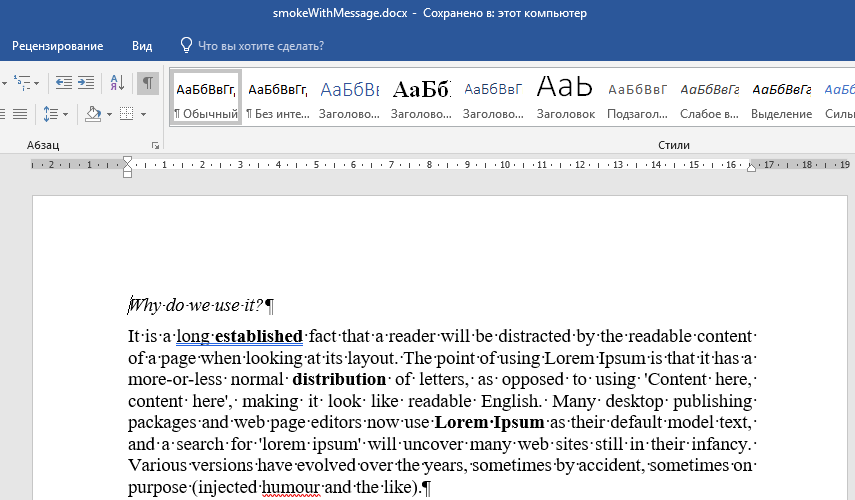


Рисунок 4.7 — DOCX-файл после внедрения информации

Теперь необходимо извлечь осажденную информацию, ранее внедренную в контейнер. Откроем контейнер, содержащий сообщение предварительно указав язык внедренного сообщения и убедимся в том, что сообщение корректно извлекается. Успешность прохождения теста представлена на рисунке 4.8.

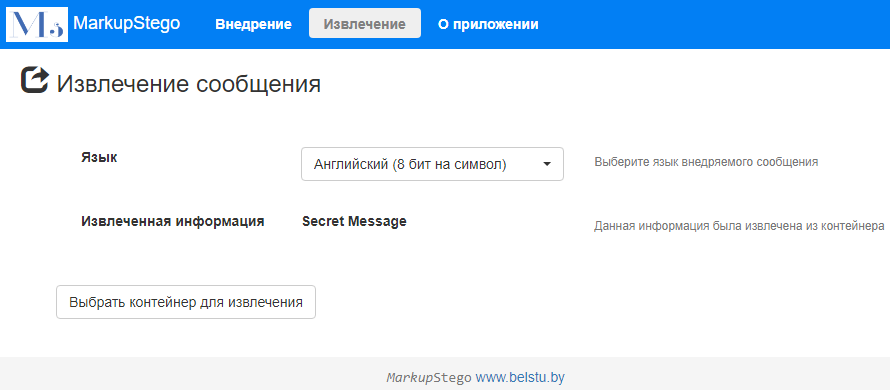


Рисунок 4.8 — Успешное извлечение осажденно информации

Как показано на рисунке 4.8 внедренная информация успешно извлечена, что свидетельствует о корректном внедрении, а также извлечении информации.

На основании проведенных действий можно удостовериться в том, что приложение прошло тестирование. Также можно с уверенностью заявить, о том, что созданный программный продукт полностью удовлетворяет тем требованиям, которые были заявлены.

4.2 Модульное тестирование

Модульное тестирование (Unit testing) — это тестирование каждой атомарной функциональности приложения отдельно, в искусственно созданной среде. Данная среда для некоторого юнита создается с помощью драйверов и заглушек.

Для написания Unit тестов в платформе .NET существует пространство имен Microsoft.VisualStudio.TestTools.UnitTesting, содержащее необходимые методы позволяющие производить сравнивание ожидаемых и полученных в результате работы методов данных. В данном пространстве имен описан класс Assert, содержащий методы необходимые для создания тестов. Для сравнения двух аргументов на равенство используется метод AreEqual, принимающий два аргумента: первый — это ожидаемое значение, а второй, то значение, которое было получено в результате работы метода. При несовпадении передаваемых аргументов будет выброшено исключение типа Microsoft.VisualStudio.TestTools.UnitTesting.AssertFailedException, которое среда разработки представляет в виде модального окна, представленного на рисунке 4.1.

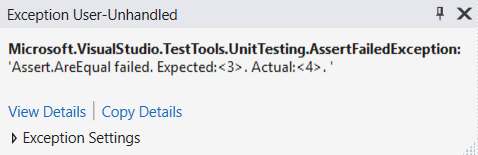


Рисунок 4.1 — Модальное окно Visual Studio сообщающее об ошибке в тесте

Если же значения совпали, то никаких исключений не выводиться, и тестовая программа завершается, а в секции «Test Explorer», представленной на рисунке 4.2 можно убедиться, что тесты выполнены успешно.

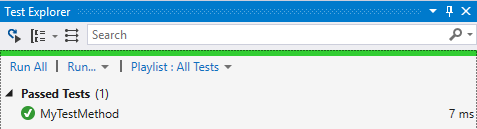


Рисунок 4.2 — Секция «Test Explorer» среды разработки Visual Studio

Данный подход, позволяет контролировать правильность работы методов, а также проверять их корректность после того, как были внесены какие-либо изменения.

Для проверки работоспособности функций выполняющих основные преобразования в приложении необходимо разработать тестовые методы, в которых вызвать проверяемые методы и сравнить результат выполнения с тем, который должен быть.

Протестируем метод ContainerCapacity вычисляющий максимальный размер сообщения которой можно разместить в контейнере. Для этого создадим тестовый метод TestContainerCapacity представленный на рисунке 4.1.

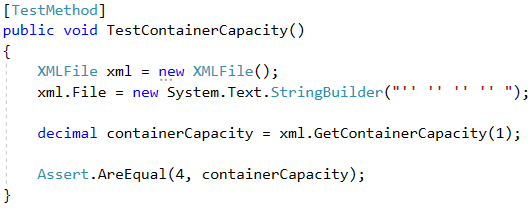


Рисунок 4.1 — Тестовый метод TestContainerCapacity

На вход тестируемому методу передается строка, состоящая из пар кавычек, на основании которой метод должен вернуть их количество. В данном случае передается 4 пары кавычек, следовательно, ожидается, что метод вернет число 4.

Следующий метод для теста — это метод MakeBinaryString, для которого создан тестовый метод TestMakeBinaryString, представленный на рисунке 4.2.

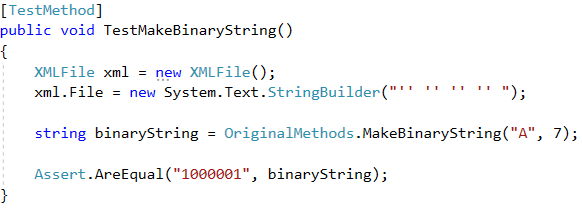


Рисунок 4.2 — Тестовый метод TestMakeBinaryString

Метод MakeBinaryString конвертирует каждый символ строки в двоичную последовательность, которая возвращается в конце его работы. В тестовом методе TestMakeBinaryString передадим на вход строку «А», двоичным представлением которой является строка «1000001».

Тест для метода EmbedMessage представлен на рисунке 4.3.

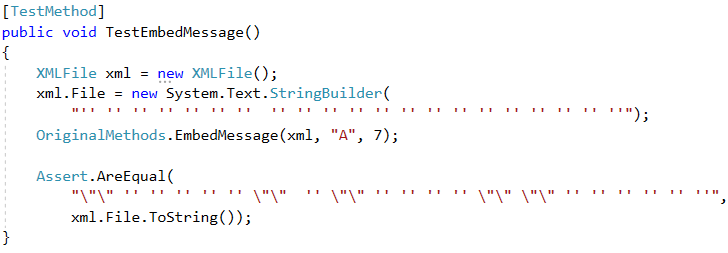


Рисунок 4.3 — Тестовый метод TestMakeBinaryString

В результате работы метода EmbedMessage модифицируется объект класса XMLFile, свойство File которого содержит текст считываемого документа. После того, как метод завершается в свойство File заноситься внедряемое сообщение в виде бинарной последовательности, путем сопоставления типов кавычек. В тесте на вход передается строка, содержащая только один тип кавычек, а в ожидаемом результате указана строка, которая содержит двойные кавычки.

Тест для метода RestoreMessage представлен на рисунке 4.4.

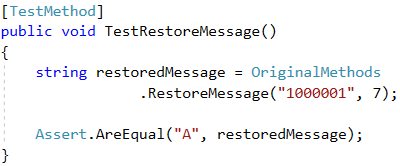


Рисунок 4.4 — Тестовый метод TestRestoreMessage

Метод RestoreMessage предназначен для конвертации двоичной последовательности в строку символов. В тесте на вход методу передается двоичная последовательность «1000001», что при переводе в символьную строку даст «А».

Тест для метода ExtractMessage представлен на рисунке 4.5.

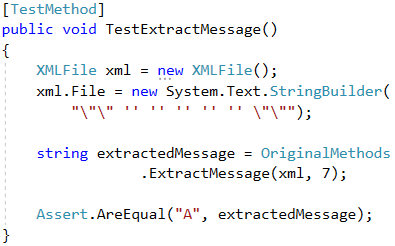


Рисунок 4.5 — Тестовый метод TestExtractMessage

Метод ExtractMessage извлекает из XML-документа, представленного в виде строки, сообщение, закодированное путем замены кавычек. В результате работы метод возвращает сообщение, которое было внедрено в XML-документ. На вход методу передаётся строка с двойными и одинарными кавычками, в которой осаждено сообщение «А», наличие которого проверяется в методе AreEqual класса Assert.

Для того, чтобы проверить соответствие входных параметров, тем результатам, которые возвращаются в результате необходимо запустить тестовые методы и проверить правильность их выполнения. Результаты запуска тестов представлены на рисунке 4.6.

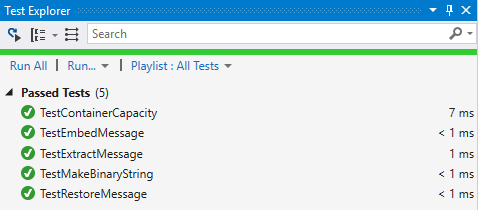


Рисунок 4.6 — Результат запуска тестов

Как видно на рисунке 4.6 все созданные тесты выполнены успешно, что гарантирует правильность работы тех методов, которые были проверены.

4.3 Вывод по разделу

Результатом выполнения данного раздела является создание тестов, для основных функций разработанного приложения. Созданные тесты гарантируют корректность работы методов осуществляющих стеганографическое преобразование контейнеров которые передаются для выполнения процедуры осаждения тайного сообщения.

Для создания тестов использовалось пространство имен Microsoft.VisualStudio.TestTools.UnitTesting, содержащее класс Assert в котором есть метод AreEqual благодаря которому можно производить процедуру сравнения ожидаемых и актуальных параметров методов.

В рамках раздела было проведено дымовое тестирование, в котором проверялась работоспособность базовых операций приложения, которые были описаны как необходимые для реализации. На основании проведенного тестирования можно сделать вывод о том, что все обязательные методы, а также элементы пользовательского интерфейса реализованы и позволяют производить процедуру осаждения\извлечения информации в стеганографический контейнер.

На основании проведенных видов тестирования можно сделать вывод о том, что программное средство стеганографического преобразования текстов-контейнеров на основе языков разметки целиком реализовано и работает согласно описанным требованиям.

5 Руководство пользователя

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

32

*ДП 05.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

Смелов В.В.

*Руководство пользователя*

Лит.

Листов

11

БГТУ 84419021, 2018

*419001, 2015*

У

4

5.1 Активация IIS сервера на операционной системе Windows

Как говорилось ранее, платформой для реализации программного продукта является ASP.NET Core. Для того, чтобы иметь возможность запустить приложение необходим сервер, в данном случае IIS (Internet Information Services).

Для активации IIS необходимо нажать правой кнопкой мыши по меню «Пуск» и зайти в раздел «Приложения и возможности». Далее кликнуть по пункту пеню «Программы и компоненты». Далее кликнуть на ссылку «Включение или отключение компонентов Windows», после чего откроется окно, в котором необходимо выставить галки так, как показано на рисунке 5.1.

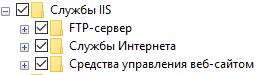


Рисунок 5.1 — Активания IIS сервера

После чего нужно нажать на кнопку «ОК» для завершения активации сервера. Далее необходимо подождать некоторое время для того, чтобы система завершила активацию сервера. После того, как система сообщит о том, что активация IIS завершена необходимо перезагрузить компьютер. После перезагрузки при указании в браузере адреса «localhost» загрузится страница с содержимым, показанным на рисунке 5.2. При корректной загрузке страницы, можно сказать, что IIS успешно активирован.

Компонент IIS включен как часть установки Windows (как для сервера, так и для рабочих станций) и требует активизации и конфигурирования. Далее представлены три способа активации IIS для различных операционных систем.

Каждая версия операционной системы Windows предлагает свою версию IIS - IIS 8 (в Windows 8), IIS 7.5 (в Windows 7) или IIS 7 (в Windows Vista). Во всех этих версиях Windows, IIS включен, но изначально не установлен.

Жигаровская С.А.

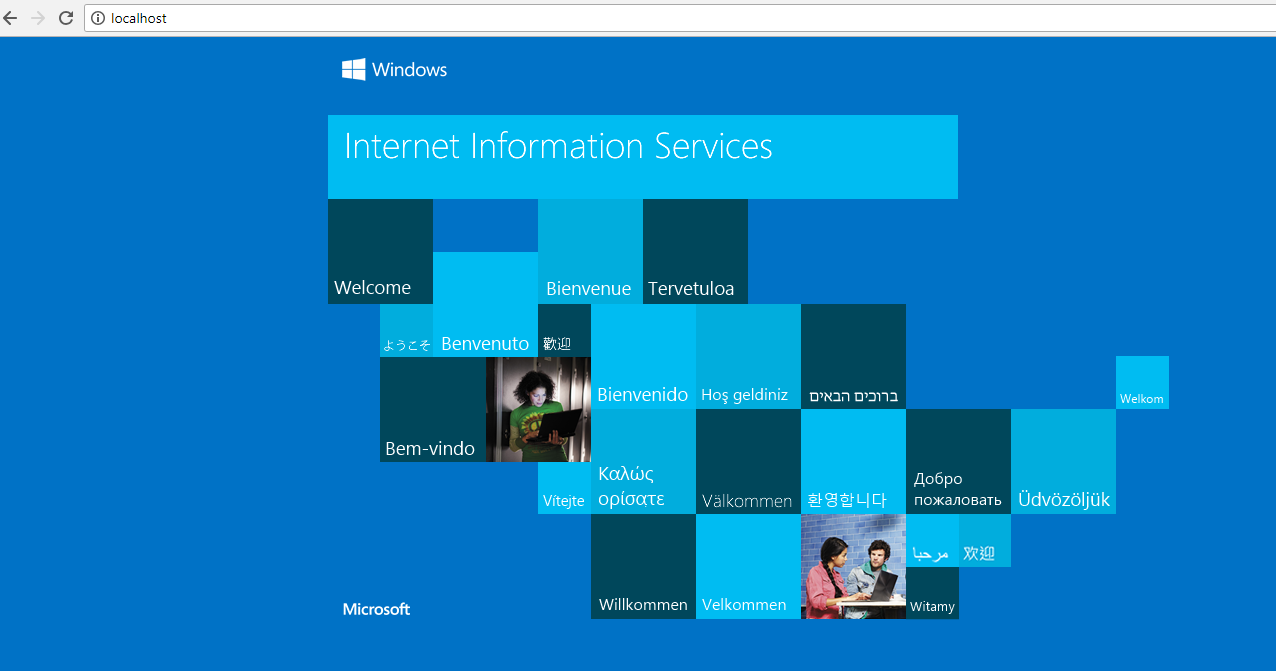


Рисунок 5.2 — Страница сайта IIS по умолчанию

5.2 Разворачивание приложения на IIS

Среда разработки Visual Studio позволяет производить процедуру разворачивания разрабатываемого приложения на сервер удобно и просто, благодаря наличию всех необходимых для этого инструментов.

Перед началом разворачивания, необходимо запустить Visual Studio от имени администратора. Для разворачивания приложения на IIS сервер необходимо нажать правой кнопкой на решение, во время того, как проект открыт в среде разработки и выбрать пункт «Publish» контекстного меню, скриншот контекстного окна представлен на рисунке 5.3.

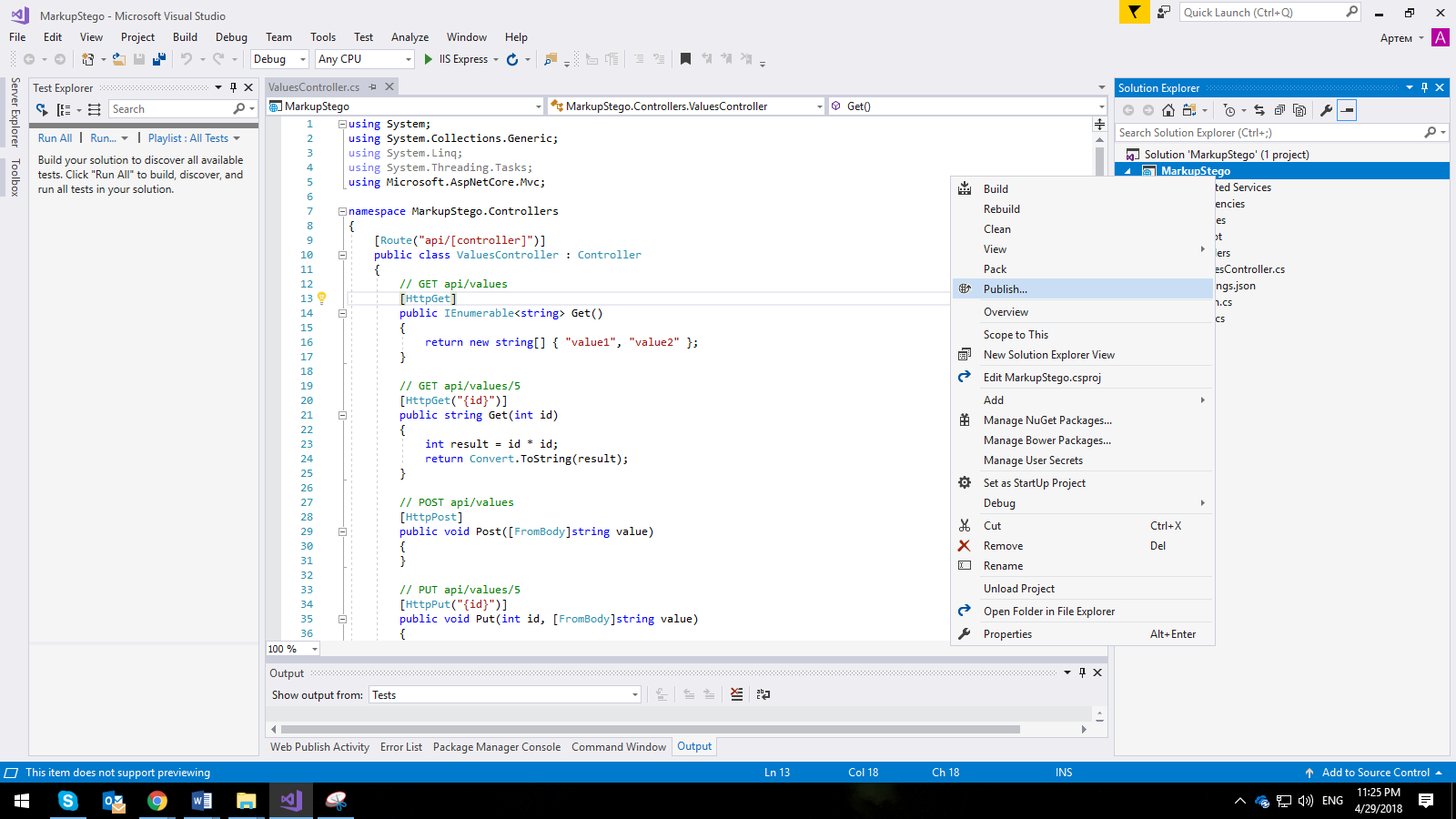


Рисунок 5.3 — Пункт «Publish» контекстного меню

Поле выбора пункта «Publish» предоставляется возможность сконфигурировать публикацию проекта, рисунок 5.4. В данном случае подойдет публикация на IIS. Активировав пункт «IIS, FTP, etc» необходимо нажать на кнопку «Publish».

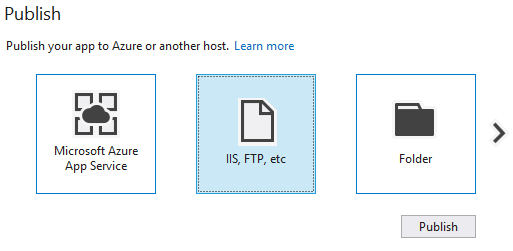


Рисунок 5.4 — Выбор пункта пеню публикация на IIS

После нажатия на кнопку «Publish», откроется окно, в котором можно указать параметры публикациии. В пункте «Publish», необходимо указать значение «File System», а в параметре «Target location» указать путь, по которому необходимо сохранить приложение.

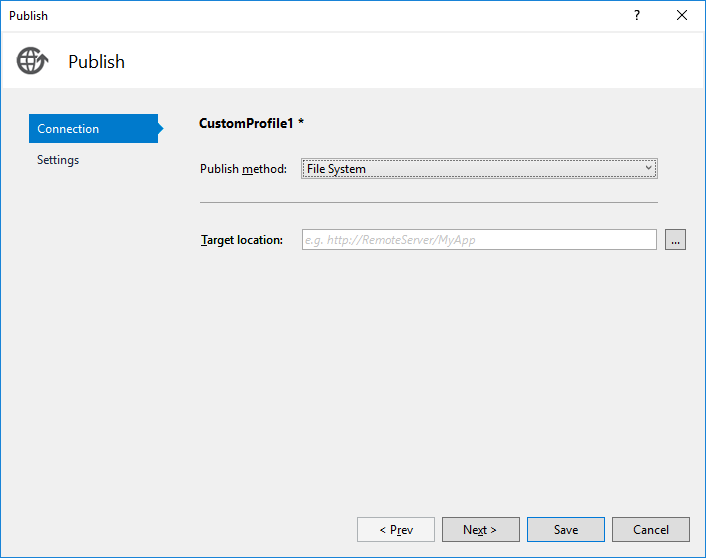


Рисунок 5.5 — Заполнение значений параметров сервера

После нажатия на кнопку «Next» появиться еще одно окно с набором конфигураций. В данном окне необходимо указать в поле «Configuration» значение «Release», а в поле «Target Framework» значение «netcoreapp2.0», после чего нажать кнопку «Save», рисунок 5.6.

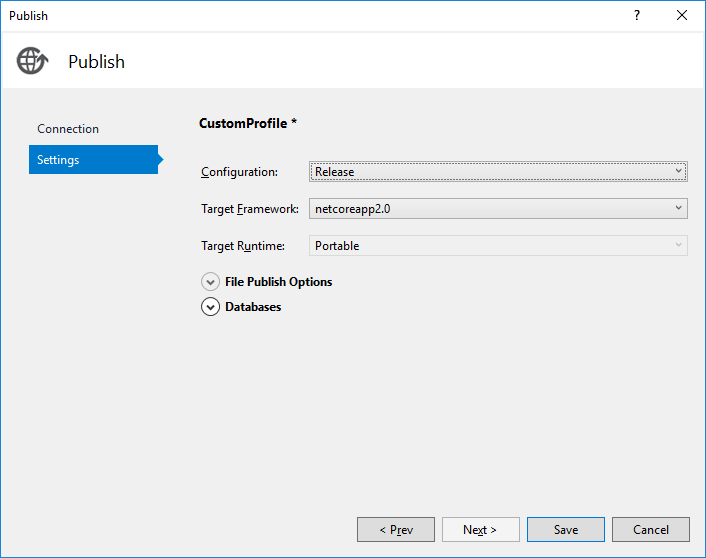


Рисунок 5.6 — Выбор целевой платформы приложения

После нажатия на кнопку «Save» профиль разворачивания сохранится и его можно будет использовать в дальнейшем. Теперь для того, чтобы развернуть приложение достаточно нажать на кнопку «Publish». После нажатия на кнопку «Publish» приложение можно найти в папке, рисунок 5.7, путь которой был указан в профиле разворачивания. Для запуска приложения, при помощи консоли, необходимо перейти в папку, в которой находиться приложение и выполнить команду «dotnet» параметром которой передать имя приложения. После выполнения команды в консоли отобразиться информация о доступе к запущенному приложениию, рисунок 5.7.

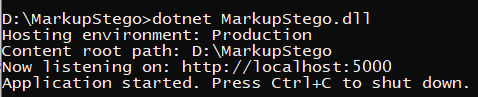


Рисунок 5.7 — Результат выполнения команды запуска приложения

5.3 Последовательность действий пользователя при осаждении/извлечении информации

Для осаждения информации с контейнер необходимо в адресной строке браузера указать путь, по которому доступно приложение, после чего пользователь сразу попадает на главной странице приложения, рисунок 5.8.

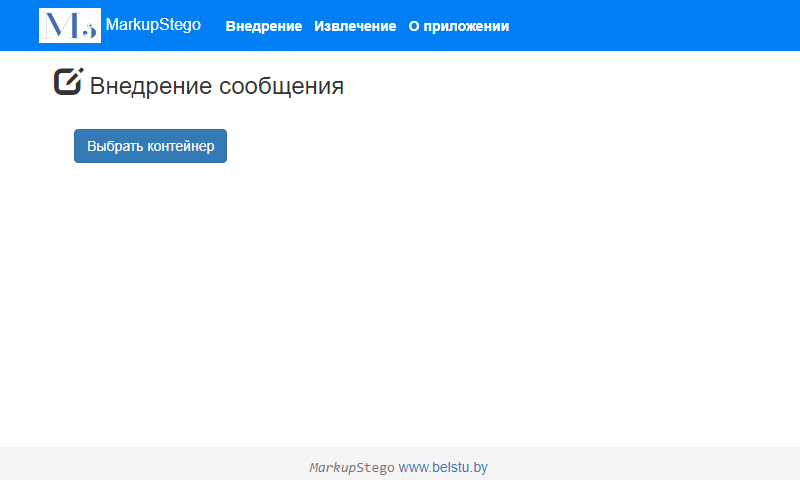


Рисунок 5.8 — Главная страница приложения «MarkupStego»

Первым шагом, является выбор контейнера, в который будет производиться осаждение сообщения. Для начала выбора контейнера необходимо нажать кнопку «Выбрать контейнер», на главной странице приложения. После нажатия, появится окно позволяющее указать файл, хранящийся на жестком диске пользователя, рисунок 5.9.

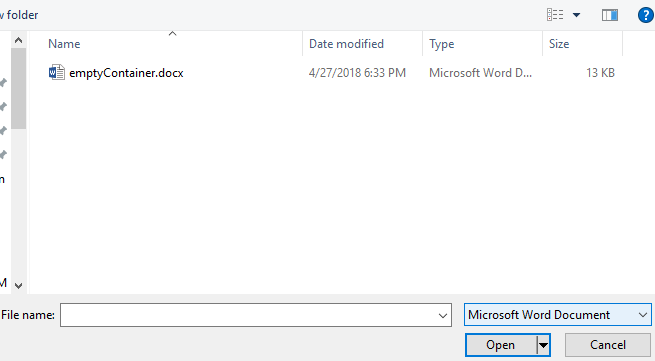


Рисунок 5.9 — Модальное окно выбора контейнера

Для того, чтобы исключить возможность загрузки файла с расширением отличным от DOCX, на выбор для загрузки предоставляются файлы только с данным расширением. После того, как пользователь завершил выбор файла, необходимо нажать на кнопку «Open», после чего файл загрузиться на сервер. При загрузке выбранного файла, сервер после его обработки высылает информацию необходимую, для успешного осаждения сообщения, которая отображается на форме, рисунок 5.10.

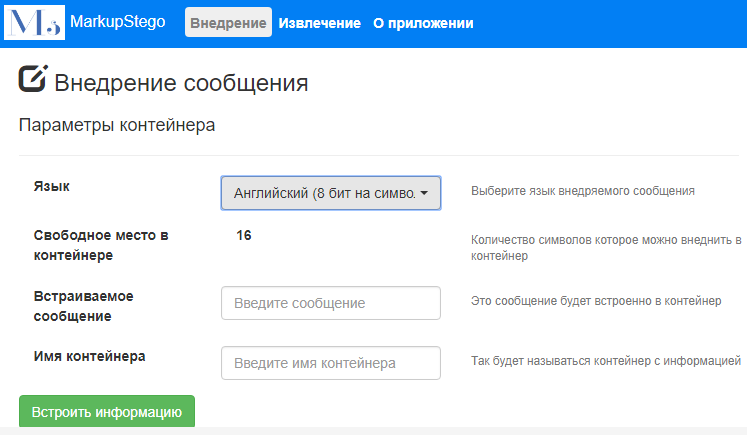


Рисунок 5.10 — Информация о пустом контейнере

Для того, чтобы осуществить внедрение в DOCX контейнер при помощи разработанного приложения, нужно указать ряд параметров. Первым параметром является язык сообщения. На выбор доступно два языка: русский и английский, рисунок 5.11.

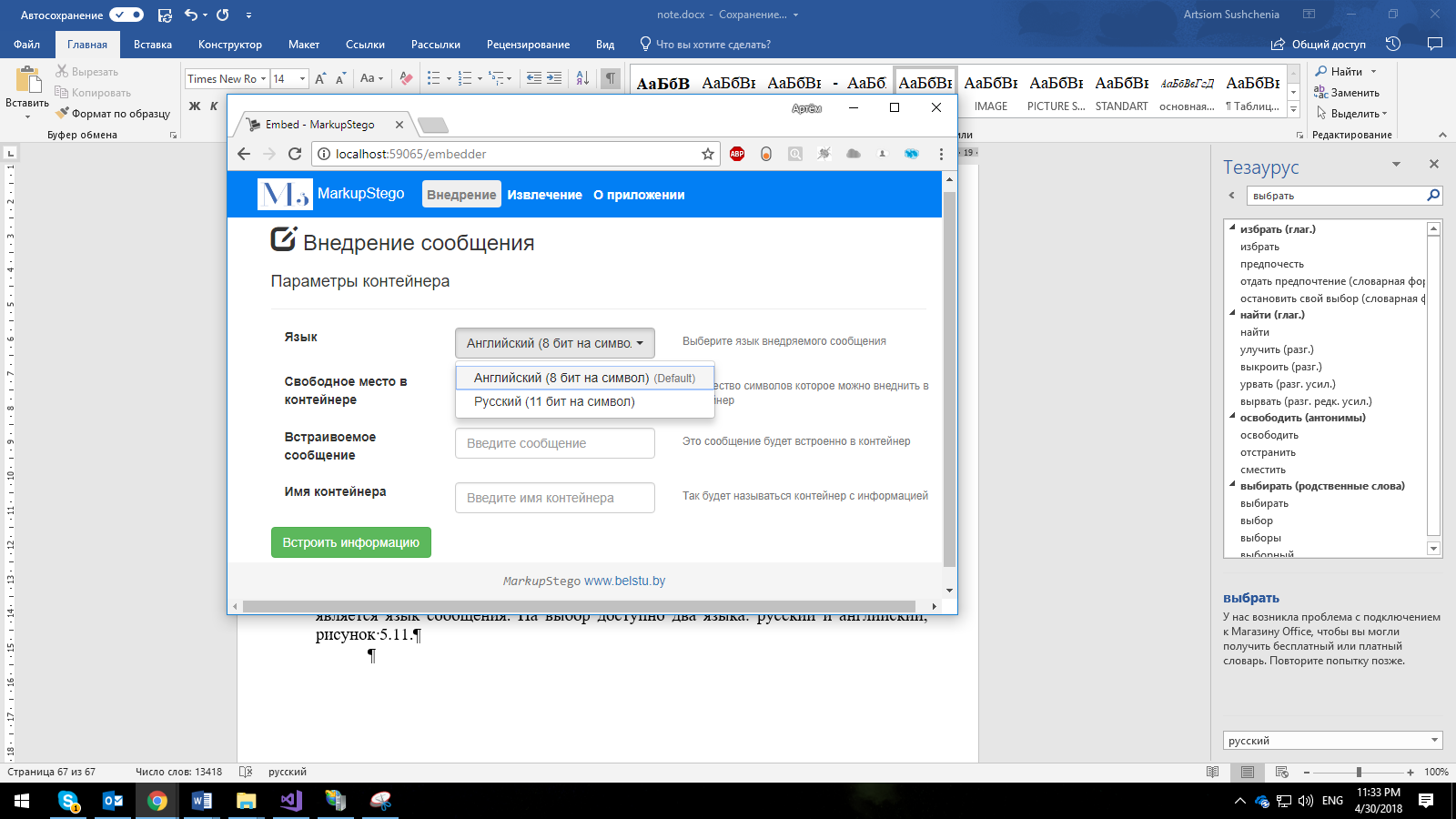


Рисунок 5.11 — Список доступных языков внедряемого сообщения

От выбора языка, зависит насколько эффективно будет использоваться размер контейнера, так как при выборе русского языка, на один символ сообщения будет отводиться 11 бит в контейнере, в то время как для английского языка можно использовать всего лишь 7 бит.

После выбора языка сообщения, необходимо ввести само тайное сообщение. Это сообщение вводиться в поле «Встраиваемое сообщение».

Следующее поле для заполнения — это поле «Имя контейнера». Именно так будет называться DOCX-документ в который будет внедрена введенная выше информация. Пример заполненной формы показан на рисунке 5.12.

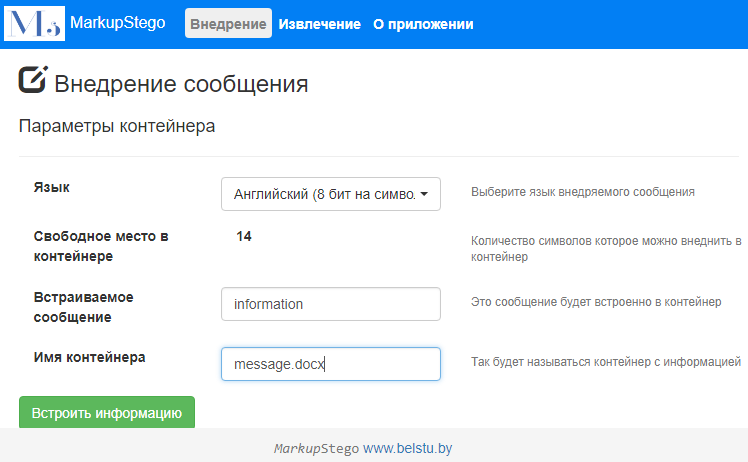


Рисунок 5.12 — Пример заполнения формы осаждения информации

После того, как форма заполнена, для завершения осаждения, необходимо нажать кнопку «Встроить информацию», после чего, появится модальное окно, которое позволяет место в файловой системе, куда будет сохранен контейнер с информацией, имеющий название, которое было указано на форме, рисунок 5.13.

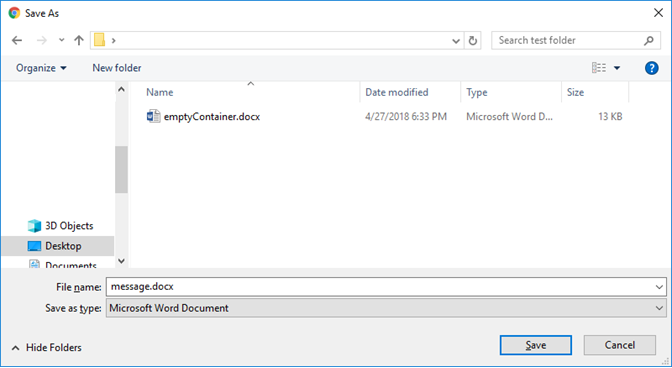


Рисунок 5.13 — Модальное окно сохранения контейнера в файловой системе

Для того, чтобы сохранить контейнер, необходимо после выбора места, нажать на кнопку «Save», после чего контейнер будет доступен по указанному пути.

При попытке осаждения сообщения, которое имеет длину больше, чем может позволить разместить контейнер, кнопка «Встроить информацию» подсвечивается другим цветом, а при наведении на нее мышью курсор изменяет свое обычное состояние, что сигнализирует о том, что встраивание информации невозможно, рисунок 5.14.

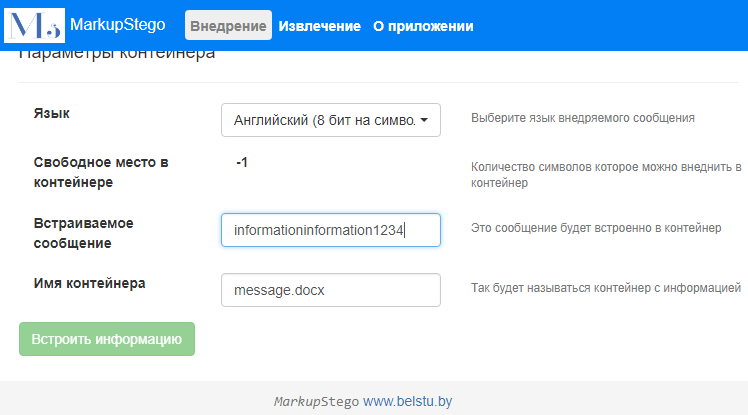


Рисунок 5.14 — Сообщение превышающее вместимость контейнера

В результате выполнения вышеперечисленных действий, пользователь получает стеганоконтейнер с расширением DOCX.

Для осуществления процедуры извлечения информации из контейнера, в который ранее, было внедрено сообщение, необходимо перейти на вкладку «Извлечение», рисунок 5.15.

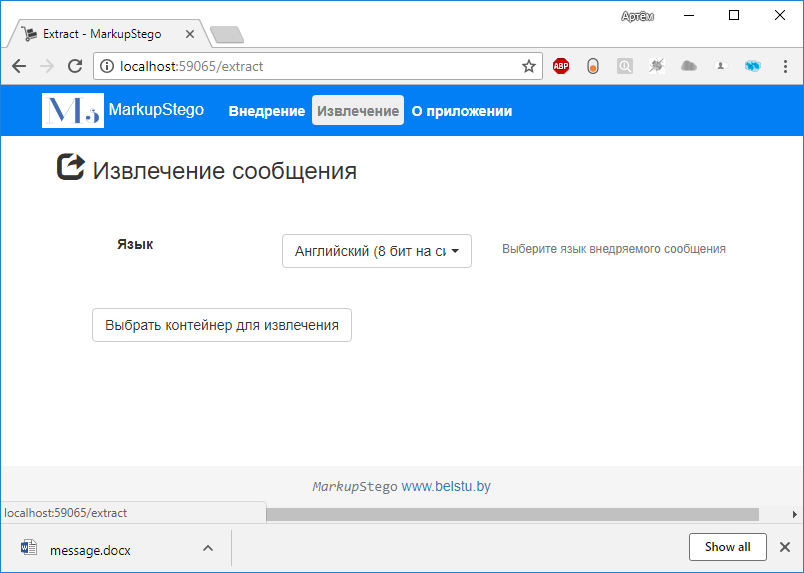


Рисунок 5.15 — Выбор вкладки «Извлечение»

После выбора вкладки «Извлечение», отобразится компонент, позволяющий произвести процедуру обратную процедуре внедрения, рисунок 5.16.

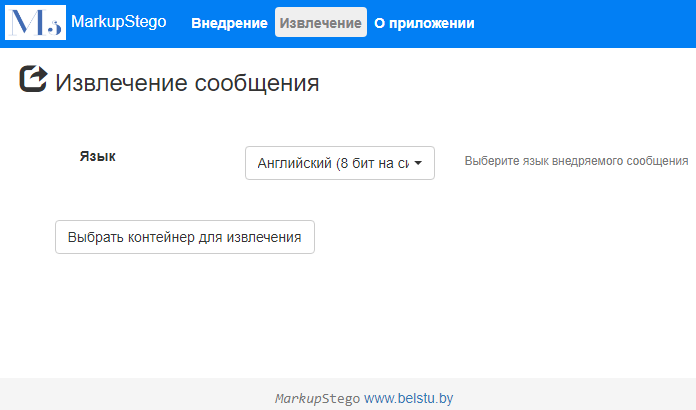


Рисунок 5.16 — Компонент производящий процедуру извлечения

Для успешного извлечения осажденного сообщения, вначале необходимо выбрать язык сообщения, которое было внедрено. Далее необходимо нажать на кнопку «Выбрать контейнер для извлечения», после чего откроется модальное окно, в котором необходимо указать документ с расширением DOCX, в который ранее было произведено осаждение, рисунок 5.17.

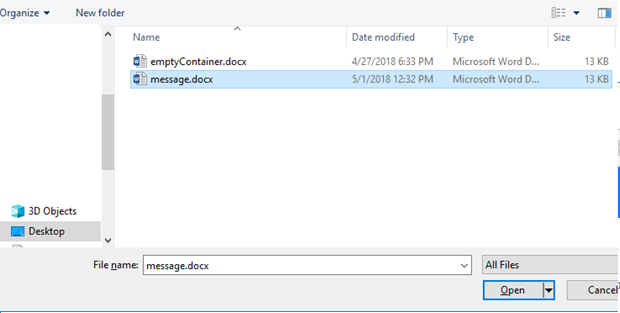


Рисунок 5.17 — Выбор контейнера для извлечения в файловой системе

После того, как выбор контейнера завершен, необходима нажать на кнопку «Open». После нажатия на кнопку «Open», выбранный файл отправится на сервер для обработки, а клиенту вернется только извлеченное сообщение, которое отобразится на форме, рисунок 5.18.

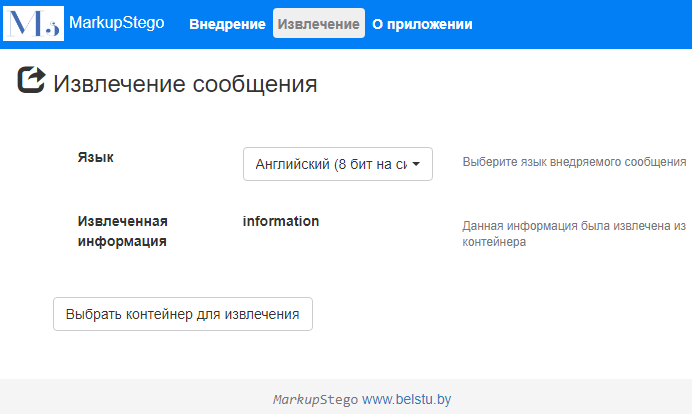


Рисунок 5.18 — Результат процедуры извлечения информации

Как видно на рисунке 5.18, внедренная ранее информация была успешно извлечена из контейнера.

Для получения информации о приложении и его авторе, необходимо перейти на вкладку «О приложении», при нажатию на которую будет отражен компонент, выводящий необходимую информацию, рисунок 5.19.

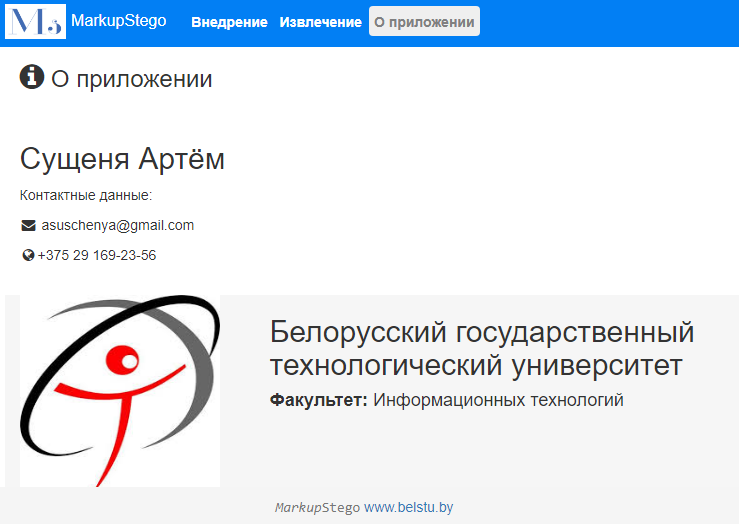


Рисунок 5.19 – Содержимое вкладки «О приложении»

5.4 Вывод по разделу

В данном разделе были продемонстрированы сценарии взаимодействия с разработанным программным средством.

Была описана процедура разворачивания приложения в операционной системе Windows. Показан пример запуска сервера приложения, а также набор действий в среде разработки Visual Studio, позволяющих произвести экспорт разрабатываемого приложения в файловую. систему, для дальнейшего использования и деплоя на сервер.

Был продемонстрирован основной функционал приложения, в который входят процедуры внедрения и извлечения информации в контейнер формата DOCX. Была пошагово расписана процедура внедрения информации, со всеми возможными вариантами пользовательского ввода.

Был продемонстрирован пример, в котором производилась процедура извлечения информации в котором, можно было убедиться, что внедряемое и извлекаемое сообщения идентичны, что свидетельствует о корректности работы программного средства.

Заключение

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

32

*ДП 00.00.ПЗ*

Разраб.

Сущеня А.А.

Провер.

.

Урбанович П.П.

Консульт.

Урбанович П.П.

Н. Контр.

Утверд.

Смелов В.В.

*Заключение*

Лит.

Листов

11

БГТУ 84419021, 2018

*419001, 2015*

У

4

В ходе выполнения дипломного проекта были рассмотрены особенности стеганографии, позволяющие использовать данную дисциплину в качестве средства сокрытия данных. На ряду с достоинствами были показаны уязвимости стеганографии, препятствующие созданию стойкого стегоконтейнера.

По теме дипломного проекта был проведен патентный поиск, показавший наличие созданных методов текстовой стеганографии. Были найдены программные средства, которые на основании разработанных алгоритмов используют в качестве контейнера для осаждения цифровые объекты. На основании проведенного исследования была поставлена цель, а также сформулированы задачи дипломного проектирования.

Разработан стеганографический метод осаждения информации в электронный документ формата DOCX, на основании метода замены кавычек в языках разметки.

Реализовано программное средство, использующее разработанный метод. Для выбора оптимальной технологии разработки были сравнены наиболее подходящие языки программирования и платформы позволяющие выполнить поставленную цель.

Для разработки программного средства была использована платформа ASP.NET Core с реализацией клиентского приложения на Angular. В качестве среды разработки была выбрана Visual Studio 2017.

Составлено руководство пользователя, в котором описаны основные сценарии приложения. Показан пример процедуры внедрения/извлечения в котором в DOCX файл была внедрена, а затем успешно извлечена тайная информация.

По теме дипломного проекта сделаны доклады на 68-й и 69-й научно-технических конференциях студентов и магистрантов, проводимых в УО «Белорусский государственный технологический университет» в 2017 и 2018 годах, а также доклад на XXI Республиканской научной конференции студентов и аспирантов. Опубликованы соответствующие материалы конференций [34].

Дипломный проект выполнен в полном объеме с реализацией всех поставленных задач.

Жигаровская С.А.

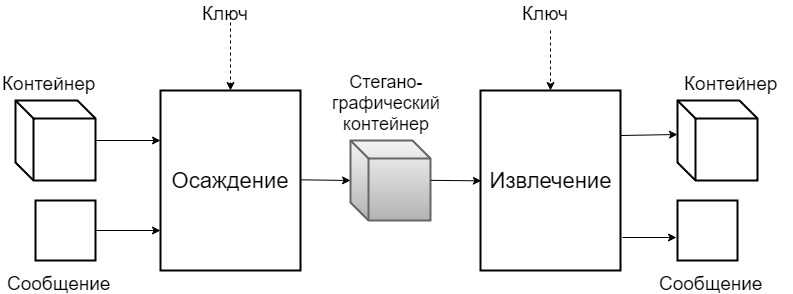


Рисунок 8 — Структурная схема стеганографической системы

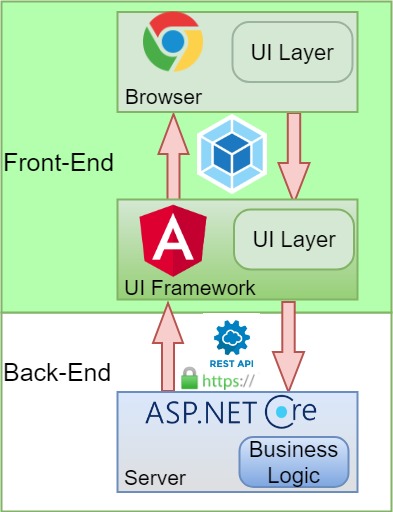


Рисунок 9 — Схема работы программного средства

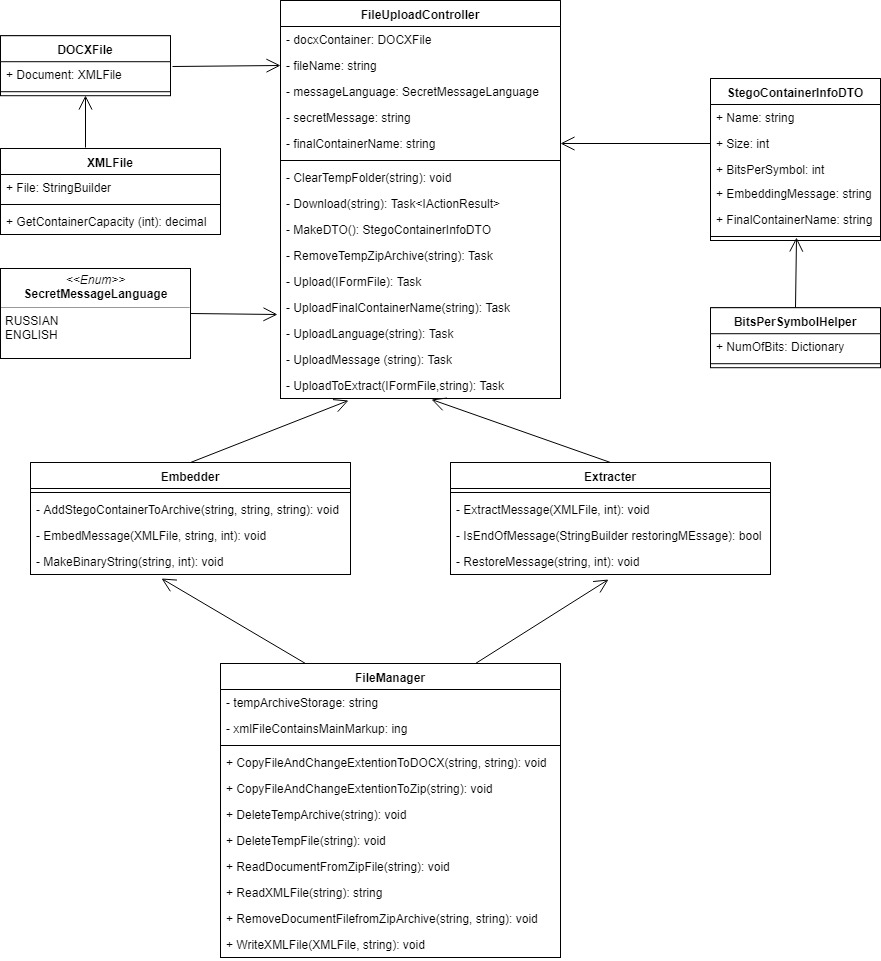


Рисунок 10 — Диаграмма классов