Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Отчет к лабораторной работе**:

«Исследование методов текстовой стеганографии»

Выполнила:

студентка 3 курса 5 группы

специальности ПОИТ

Кубик Е.В.

Минск 2021

**1. Классификация, сущность и основные особенности базовых методов текстовой стеганографии**

Было отмечено, что к текстовой стеганографии относятся методы, предусматривающие использование в качестве контейнера файла-документа текстового типа.

Многообразие методов текстовой стеганографии подразделяется на синтаксические методы, которые не затрагивают семантику текстового сообщения, и лингвистические, которые основаны на эквивалентной трансформации текстовых файлов-контейнеров, сохраняющей смысловое содержание текста, его семантику.

Для понимания сущности некоторых из методов полезно познакомиться с важнейшими особенностями и параметрами использования стилей (в т. ч. – пространственно-геометрическими параметрами шрифтов), на основе которых строится текстовый файл-контейнер. На рис. 11.1 показаны основные из параметров шрифта.



Рисунок 15.1 – Параметры шрифта

К синтаксическим методам компьютерной стеганографии, которые характеризуются сравнительно невысокой эффективностью (с точки зрения объема осаждаемой информации) относятся следующие (такие методы мы отнесем к числу базовых синтаксических методов):

* изменение расстояния между строками электронного текста (Line-Shift Coding); называется методом изменения межстрочных интервалов; сущность заключается в том, что используется текст с различными межстрочными расстояниями: выделяется максимальное и минимальное расстояния между строками, позволяющее кодировать соответственно символы «1» и «0» осаждаемого сообщения;
* изменение расстояния между словами в одной строке электронного текста (Word-Shift Coding); суть метода состоит в том, что осаждение информации основано на модификации расстояния между словами текста-контейнера;
* изменение количества пробелов между словами (частный случай метода Word-Shift Coding); Основан та том, что, например, чередование одинарного пробела и двойного (хх­\_хх\_\_хх) кодирует «1», переход же с двойного пробела на одинарный кодирует «0» (хх\_\_хх\_хх);
* на основе внесения специфических изменений в шрифты, т. е. начертания отдельных букв (Feature Coding); заключается в изменении написания отдельных букв используемого стандартного шрифта: визуально заметны различные образы, соответствующие буквам с верхними (например, l, t, d) или нижними (например, a, g) выносными элементами (см. рис. 13.1); например, букву «А» можно модифицировать, незначительно укорачивая длинную нижнюю часть буквы (см. рис. 1.2);

а) пустой контейнер; б) заполненный контейнер

Рисунок 15.2 - Пример применения метода Feature Coding

* изменение интервала табуляции; аналогичен вышеописанному методу изменения количества пробелов, только в этом случае меняется не количество пробелов, а соответственно расстояние между строками и интервал табуляции;
* Null Chipper (дословно – несуществующий, нулевой лепет); предполагает размещение тайной информации на установленных позициях слов или в определенных словах текста-контейнера, который, как правило, лишен логического смысла (как видно, действительно лепет);
* увеличение длины строки; предусматривает искусственное увеличение длины каждой строки за счет пробелов: например, нет пробела (определяется положением знака перехода на новую строку) – «0», один пробел – «1» (рис. 13.3);
* использование регистра букв; для обозначения бита секретного сообщения, представленного единицей, используется символ нижнего регистра, а нулем — верхнего (или наоборот);



Рисунок 15.3 - Пример реализации метода увеличения длины строки

* использование невидимых символов; знак пробел кодируется символом с кодом 32, но в тексте его можно заменить также символом, имеющим код 255 (или 0), который является «невидимым» и отображается как пробел.

Рассмотренные базовые методы могут применяться независимо и совместно, сохраняют исходный смысл текста, а обеспечиваемые ими показатели плотности кодирования при совмещении складываются.

Еще одна важная особенность. Перечисленные методы работают успешно до тех пор, пока тексты представлены в коде ASCII.

Методы также легко применяются к любому тексту, независимо от его содержания, назначения и языка. Синтаксические системы стеганографии легко реализуются в программном коде, так как они полностью автоматические и не требуют вмешательства оператора. Однако синтаксические методы неустойчивы к форматированию текста (вспомним робастность систем на основе ЦВЗ), и поэтому информация может быть потеряна при простом применении иного стиля форматирования текста-контейнера, скрывающего в себе стегосообщение. К тому же с помощью синтаксических методов можно передать незначительное количество информации.

Существуют также стеганографические методы, которые интерпретируют текст как двоичное изображение. Необходимо отметить, что данные методы нечувствительны к изменению масштаба документа, что обеспечивает им хорошую устойчивость к большинству искажений, которые могут иметь место при активных атаках.

К числу основных лингвистических методов относятся:

* метод синонимов;
* метод переменной длины слова; основан на том, что длина слов в сообщении зависит от содержания секретного сообщения и способа кодирования слов: обычно одно слово текста-контейнера определенной длины кодирует два бита информации из стеганосообщения; например, слова текста длиной в 4 и 8 символов могут означать комбинацию бит «00», длиной в 5 и 9 — «01», 6 и 10 — «10», 7 и 11 букв — «11»; слова короче 4 и длиннее 11 букв можно вставлять где угодно для лексической и грамматической связки слов в предложении – программное приложение, которое декодирует принятое сообщение (извлекает сообщение из стеганоконтейнера), будет просто игнорировать их;
* метод первой буквы – программа-помощник в этом методе накладывает ограничение уже не на длину слова, а на первую (можно на вторую) букву; обычно одну и ту же комбинацию могут кодировать несколько букв, например, комбинацию «101» означают слова, начинающиеся с «А», «Г» или «Т;
* мимикрия; мимикрия генерирует осмысленный текст, используя синтаксис, описанный в Context Free Grammar (CFG), и встраивает информацию, выбирая из CFG определенные фразы и слова; грамматика CFG – это один из способов описания языка, который состоит из статических слов и фраз языка, а также узлов.

**2. Методы текстовой стеганографии на основе модификации цветовых и пространственно-геометрических параметров символов текста-контейнера**

Поддерживаемые форматы документов-контейнеров, которые мы хотим рассматривать как объекты для скрытия тайной информации – любые, способные хранить цвет и иные указанные параметры символов и которые можно открыть с помощью процессора MS Office Word: (\*.doc, \*.docx), \*.rtf (межплатформенный формат хранения размеченных текстовых документов), \*.odt (открытый формат документов для офисных приложений).

Что касается реализации метода на основе модификации цвета символов текста-контейнера, по сути своей он схож с классическим методом наименее значащих бит и опирается на использовании цветовой модели RGB.

К основным пространственно-геометрическим параметрам символов (и текста в целом) относятся, которые могут быть использованы как инструменты для стеганографического преобразования, относятся апрош и кернинг.

**3. Метод на основе апроша**

Апрош определяет расстояние между соседними символами текста. Фактически апрош состоит из двух таких расстояний – полуапрошей, являющихся как бы пространством, прилегающим к каждому из символов-соседей (см. рис. 1.1 и 1.4). Мы далее будем обращаться только к апрошу. Согласно существующим техническим правилам набора нормальный апрош должен быть равен половине [кегля](http://slovari.yandex.ru/%7E%D0%BA%D0%BD%D0%B8%D0%B3%D0%B8/%D0%98%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B9%20%D1%81%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/%D0%9A%D0%B5%D0%B3%D0%BB%D1%8C%20%28%D0%BA%D0%B5%D0%B3%D0%B5%D0%BB%D1%8C%29/) (размера) шрифта.

Идея метода заключается в следующем. Встраивание сообщения в контейнер может быть основано на модификации базового (устанавливаемого текстовым процессором по умолчанию) значения апроша, ао, его изменением от базового до некоторого максимального аmax (или минимального), аmin, которое зрительно не должно отличаться от стандартного. Такое изменение производится с определенным шагом (дискретно) Δаi, каждому значению которого присваивается определенный бит или определенная комбинация бит.



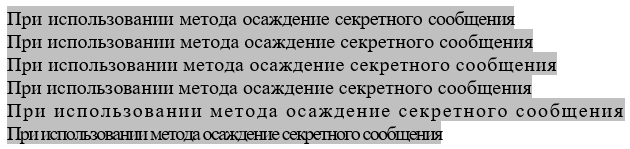
Рисунок 15.4 - Измерение апроша

Изменение величины апроша между двумя определенными символами текста относительно базового значения ао на небольшое расстояние (пункты (пт) или доли пункта) формально можно представить в следующем виде:

at = ао + Δаt . (1.1)

Такое изменение не должно вызывать визуально заметного уплотнения (Δаt < 0) или разрежения (Δаt > 0) групп символов. В текстовом процессоре MS Word апрош может принимать значения в диапазоне от 0 до 1584 пунктов.

Для примера и визуального представления об особенностях установки данного пространственно-геометрического параметра шрифта на рисунке 1.5 приведена строка текста с различными параметрами апроша.

Рисунок 15.5 - Примеры использования различного апроша

На этом рисунке вторая строка оформлена с использованием стандартного (обычного) апроша. В первой строке применено уплотнение во всех словах, кроме первого: во втором слове – на 0,1 пт, в третьем – на 0,2 пт, в четвертом – на 0,3 пт, в пятом – на 0,4 пт, в пятом – на 0,5 пт; в третьей строке слова со второго по пятое оформлены с изменением Δа в противоположную сторону, т.е. с разрежением. В четвертой строке применялось уплотнение (отрицательный Δа) или разрежение (положительный Δа) лишь к отдельным символам первого («При») и второго («использовании») слов: для «П» – Δа = –0,1 пт; «ри» – Δа = –0,2; «и» – Δа = 0; «с» – Δа = 0,1; «п» – Δа = 0; «о» – Δа = –0,1; «л» – Δа = –0,2; «ь» – Δа = –0,3; «з» – Δа = 0,4; «ов» – Δа = –0,3; «ан» – Δа = 0,2; «ии» – Δа = 0. Пятая строка целиком отформатирована при Δа = 1 пт, а шестая – при Δа = –1 пт.

Особенностью рассматриваемого метода является возможность одноразового размещения (в апроше одного символа) числа бит, определяемого дискретной разницей между минимальным и максимальным значениями Δа. Например, если отсчет вести от Δамин до установленного интервала Δаt в виде параметра 0,1\*nд (пт), то количество условных дискретных единиц nд, представленное в бинарном виде, определяет число бит, которые можно таким образом разместить; например, Δамин = –0,5 пт, а Δаt = 0,3 пт. Разница между этими величинами составляет 0,8 пт: 8\*0,1 или nдt = 8 ( в двоичном виде – 1000; в первом приближении именного такую бинарную комбинацию можно разместить (осадить) путем модификации конкретного апроша). На этой основе могут быть разработаны различные варианты кодировки осаждаемых комбинаций.

**4. Метод на основе кернинга**

В текстовых документах встречаются такие сочетания знаков, которые образовывают визуальные «дыры» либо «сгущения». Например, в текстах на основе кириллицы – это такие сочетания: «ГА», «TA», «ATA», «ЬТ» и т. п., на основе латиницы – «AY», «AV», «T;», «ff», а на основе греческого алфавита – «ΘΑ», «ΔΟ», «λκ» и др. Такие сочетания называются кернинговыми парами. Особенности «кернингования» приведены на рис. 1.6.

Под кернингом обычно понимается процесс изменения межсимвольного расстояние между отдельными парами символов или кернинговыми парами (именно фактор парности отличает кернинг от апроша).

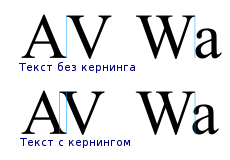


Рисунок 15.6 - Пояснение к понятию кернинга

Таким образом, технология кернинга, появившаяся в полиграфии после внедрения фотонабора (а затем и компьютерного набора), включает подбор межбуквенных интервалов для конкретных пар букв с целью улучшения внешнего вида и удобочитаемости текста. Такой избирательный подбор позволяет компенсировать неравномерности визуальной плотности текста, получаемой при использовании стандартных апрошей для каждой буквы.

Очевидно, что промежуток между «A» и «V» в первой строке на рис. 1.6 гораздо больше, чем во втором, хотя формально они одинаковы. В данном случае сочетание «A V» как раз и является кернинговой парой. После применения кернинга визуальное восприятие текста улучшилось.

С появлением цифрового фотонабора стало возможным хранить такие критические сочетания знаков (кернинговые пары) для некоторого условного шрифта, общее число которых мы обозначим Nk, в памяти компьютера с указанием величины (σ), на которую необходимо сдвинуть символы, чтобы визуально выровнять буквенные просветы.

Как правило, текстовые редакторы или процессоры содержат встроенные средства настройки кернинга, который определяет стандартный межбуквенный интервал для того или иного шрифта. При этом σ устанавливается в соответствии со значениями из таблицы кернинговых пар, встроенной в вышеуказанный [файл со шрифтом](http://infogra.ru/typography/fonts). Такая настройка позволяет выровнять шрифт и является стандартной. В некоторых шрифтах сейчас количество пар доходит до нескольких тысяч.

Значение кернинга может быть как положительным (когда знаки раздвигаются, σ > 0), так и отрицательным (когда сдвигаются, σ < 0). Эта величина в программах верстки устанавливается в процентах от ширины символа пробела используемого шрифта.

Следуя общепринятой методике, кернинг, как параметр, будем измерять в тысячных долях круглой шпации (Еm) — единицы измерения, которая определяется относительно текущего размера шрифта и равна ширине символа «М». Например, для шрифта размером 6 пунктов 1 круглая шпация равна 6 пунктам, а для шрифта размером 10 пунктов — 10 пунктам. Таким образом, размер кернинга, σ, строго пропорционален текущему размеру шрифта. Сдвиги букв относительно автоматически установленного межсимвольного расстояния (измеряемого, например, апрошем) можно производить с различным шагом: от 0,01 до 0,04 величины Еm, в зависимости от нужной точности.

Как следует из анализа общих принципов задания размера кернинга и управления этим размером, предлагаемый метод основывается на принудительном применении кернинга, не зависящем от установок параметров текста-контейнера, созданных средствами текстового процессора или иного специализированного текстового редактора. Здесь есть одна важная особенность. Изменять значения кернинга можно лишь в программах верстки (например, таких, как InDesign) или в программах, предназначенных для работы с векторной или растровой графикой (например, CorelDraw, Photoshop). В текстовом процессоре MS Word значения кернинга автоматически установлены в таблицах кернинговых пар каждого шрифта, доступа к которым нет.

Здесь возможности пользователя практически связаны лишь с указанием минимального размера шрифта, для которого можно применять кернинг. Это означает, что, если текст набран на основе шрифта размером, например, 14 пт, а мы в специальной опции (Главная/Шрифт) установили минимальный размер в 15 пт, при котором будет выполняться кернинг (см. поясняющую иллюстрацию на рис. 1.7), то для нашего текста эта процедура (иногда ее называют «кернингованием») не будет выполнена.

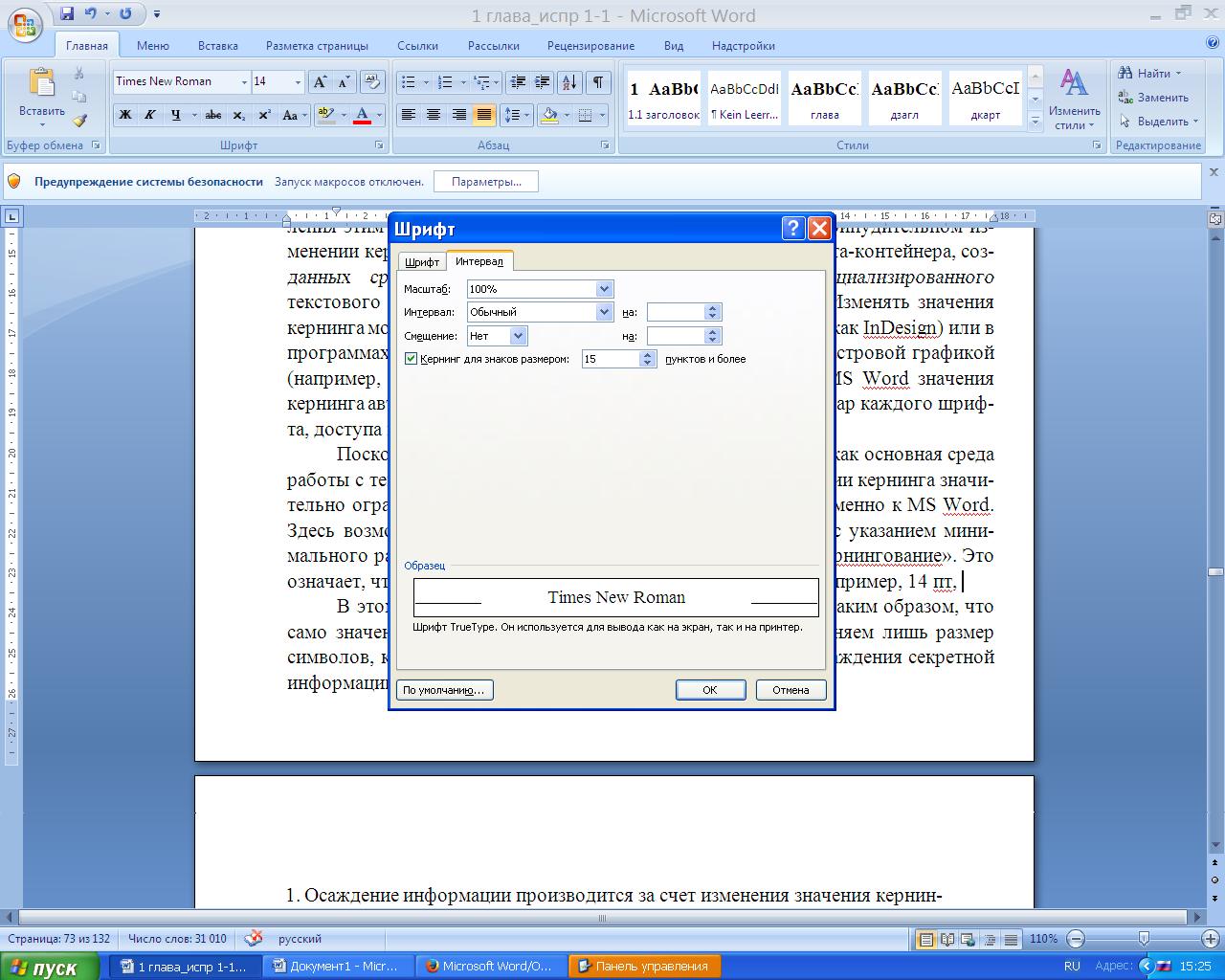


Рисунок 15.7 - Пример установки параметров для применения кернинга

Таким образом, рассматриваемый метод может быть интерпретирован так, что само значение кернинга мы программно не изменяем, а изменяем лишь размер символов, к которым будет применен кернинг в результате осаждения секретной информации.

Следует принять во внимание также следующую важную особенность. Текстовый процессор MS Word позволяет изменять параметры кернинга для знаков определенного размера. Размер символов, к которым может быть применим кернинг, должен быть в диапазоне между 1 и 1638 пт; в частном случае этот размер может быть указан в последней строке окна на рис. 1.6 (а сам размер шрифта можно устанавливать с точностью до 0,5 пт).

Заметим, что выбор символов текста для осаждения сообщения на основе модификации цветовых и пространственно-геометрических параметров может выполняться на основе ранее описанных принципов в соответствии с одним из элементов ключевой информации:

* локальный разброс;
* случайный (псевдослучайный) разброс;
* случайный (псевдослучайный) разброс с памятью или подряд.

**Практическая часть**

Приложение написано на языке программирования C# и позволяет зашифровать и расшифровать стеганографический текст.

Также приложение позволяет выполнить операций зашифрования/ расшифрования и вывести исходной текст и соответствующий ему зашифрованный текст в файлы, чтобы в последующем оценить стеганографическую криптографию по тексту

На рисунке 2.1 показана реализация алгоритма сохранения информации методом модификации апроша.

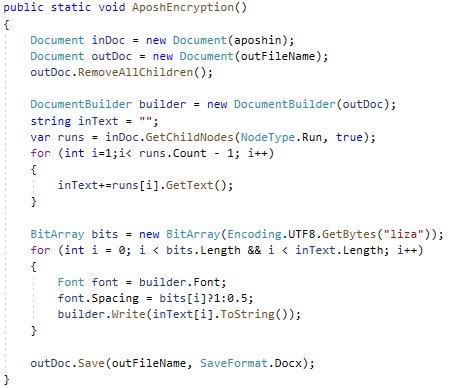


Рисунок 2.1– Модификация апроша

Сперва получается текст-контейнер, в данном случае читается данный текстовый файл. Сохраняемое сообщение преобразуется в набор битов. При переборе всех символов контейнера каждому символу задается значение апроша в зависимости от значения соответствующего бита при этом символ с заданным апрошем сохраняется в выходной файл. Обход происходит пока не закончится текст контейнера либо не закончится секретное сообщение. Таким образом на каждый бит секретного сообщения приходится один символ контейнера, однако эту отношение можно увеличить ставя не 2 варианта размера апроша, а больше.

На рисунке 2.2 показывается реализация получения секретного сообщения из контейнера.

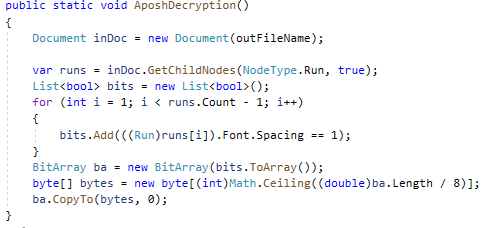


Рисунок 2.2 – Получения секретного сообщения

Для расшифрования обходится каждый символ исходного текста и берется значение его апроша. в данном случае максимальное значение апроша соответствует значению бита равным единице, при минимальном апроши бит равняется нулю. Таким образом из каждого символа получается бит сообщения, после чего биты состоят и из них получают текст сообщения

Для помещения сообщения используется подобный алгоритм как и с апрошем символов, однако в этом случае изменяется только длина строки путем добавления пробела в конец. На рисунке 2.3 показывается реализация модификации длины строки.

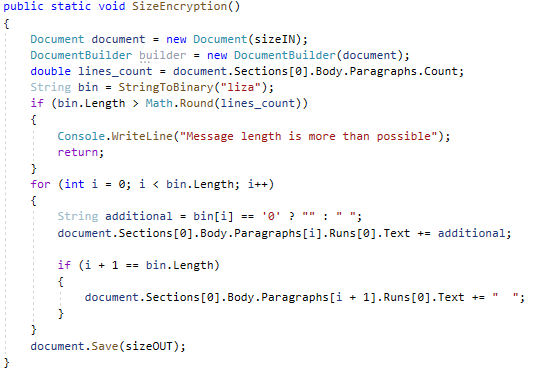


Рисунок 2.3 – Реализация модификации расстояния между строками

### Символы контейнера перебираются, после чего, при нахождении символа пробела в конце берет его согласно текучему шифруемому биту сообщения. Таким образом контейнер должен быть в разы больше чем при модификации без добавлении пробела.

Алгоритм снова повторяет соответствующий алгоритм на основе модификаций апроша, однако в данном случае учитываются только пробелы, где снова минимальный апрош означает нулевой бит а максимальный – единичный.

На рисунке 2.4 показывается исходный контйнер.

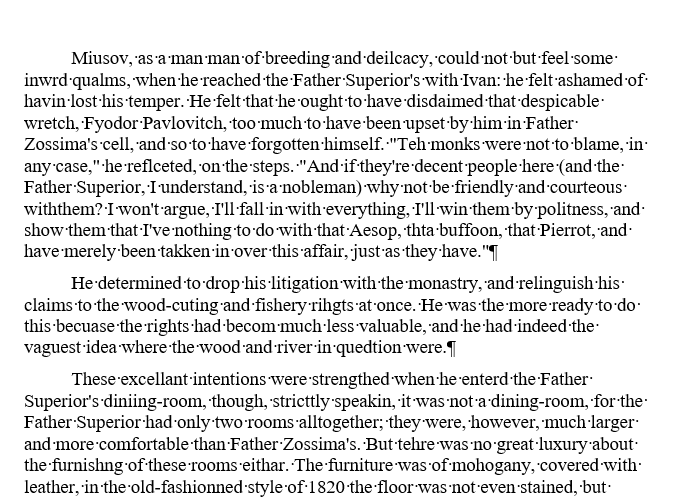


Рисунок 2.4 – Исходный контейнер

Контейнер должен быть большим чтобы вместить все символы.

На рисунке 2.5 показывается заполненный контейнер.

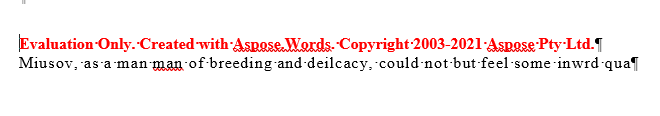


Рисунок 2.5 – заполненный контейнер апроша

Можно заметить что одинаковые слова на разных строках занимают разную длину.

На рисунке 2.6 показывается заполненный контейнер с разными длинами строк.

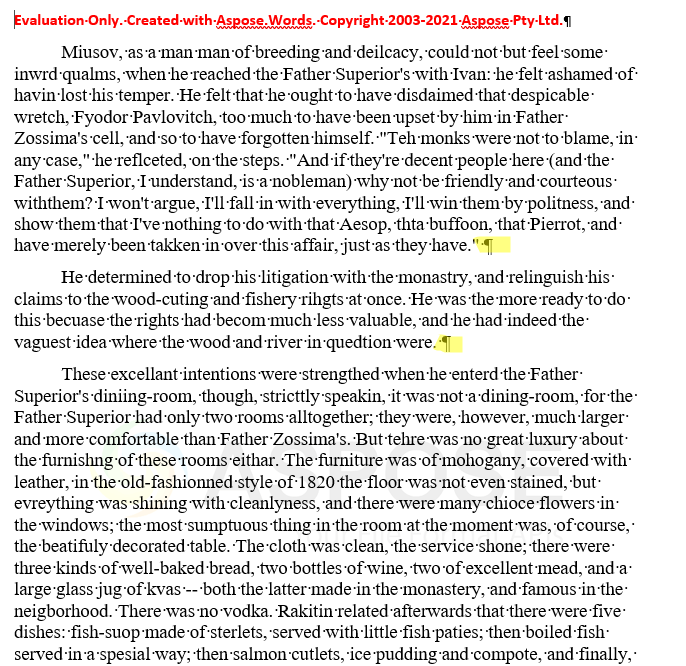


Рисунок 2.6 – Заполненный контейнер с различными длинами строк

В этом случае длины слов не различаются, однако длины строк различны из-за пробела в конце строки.

**Вывод**

В данной лабораторной работе я закрепила теоретические знания из области текстовой стеганографии, классификации, моделирования стеганосистем подобного вида и сущности основных методов. Изучила основные алгоритмы осаждения/извлечения тайной информации на основе методов текстовой стеганографии. А также познакомилась с методиками оценки стеганографической стойкости методов.