МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Лабораторная работа № 15**

**ИССЛЕДОВАНИЕ МЕТОДОВ ТЕКСТОВОЙ СТЕГАНОГРАФИИ**

Разработал: Бай И.О.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Цель:** изучение стеганографических методов встраивания/извлечения тайной информации с использованием электронного файла-контейнера текстового формата, приобретение практических навыков программной реализации методов.

1. **Теоретические сведения**

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.

К синтаксическим методам компьютерной стеганографии, которые характеризуются сравнительно невысокой эффективностью (с точки зрения объема встраиваемой 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; например, букву «А» можно модифицировать, незначительно укорачивая длинную нижнюю часть буквы;

• 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»;

• использование регистра букв; для обозначения бита секретного сообщения, представленного единицей, используется символ нижнего регистра, а нулем – верхнего (или наоборот);

• 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; в качестве примера приведем подмножество синонимов: {«тайный», «секретный», «конфиденциальный», «доверительный»}. В приведенном подмножестве каждое слово имеет единственное одинаковое смысловое значение, что позволяет закодировать каждое слово своим уникальным кодом (т. е. выполнить операцию осаждения), например, «доверительный» – 00, «конфиденциальный» – 01, «секретный» – 10, «тайный» – 11. Подобное кодирование позволяет выбирать одно из четырех слов (как видим, они для удобства расположены по алфавиту) в зависимости от двух битов секретного сообщения. Отметим, что при этом, независимо какое из четырех слов будет выбрано, семантика сообщения не изменится. Очевидно, что при этом количество символов, соответствующих одному из синонимов используемого подмножества, зависит от общего числа элементов в подмножестве. Кроме того, обеим сторонам стеганосистемы должен быть известен общий алгоритм кодирования, т. е. один из ключей системы. Следует отметить, что в каждом подмножестве синонимов их упорядочивание должно выполняться по одному и тому же алгоритму и у отправителя сообщения, и у его получателя. В случае наличия слов с несколькими смысловыми значениями подобное кодирование оказывается невозможным. Также невозможно кодирование, если один из синонимов состоит из двух (или более) разделенных пробелом слов;

• 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 битов «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.

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 битов и опирается на использовании цветовой модели RGB.

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

1. **Практическая часть**

Для модификации цвета и межстрочного расстояния было разработано два приложения.

В первом случае цвет символов в процессе встраивания меняется со случайным шагом на заранее определённые цвета в зависимости от того, 0 или единицу необходимо зашифровать. Здесь единице соответствует синий цвет, нулю – красный, пробелу, если такой есть в исходном сообщении, – голубой.

Код приведён в листинге 15.1

|  |
| --- |
| encrypt\_word = ConvertMessToByte(ConvertToASCII(Word.Text));  Document doc = new Document();  Paragraph paragraph1 = doc.AddSection().AddParagraph();  paragraph1.AppendText(text);  int part = text.Length / encrypt\_word.Length;  if (part > 1)  {  TextRange textRange = new TextRange(doc);  for (int i = 0; i < encrypt\_word.Length; i++)  {  var randPositin = random.Next(part \* i + 1, part \* (i + 1));  TextSelection selections = new TextSelection(paragraph1, randPositin, randPositin + 1);  textRange = selections.GetAsOneRange();  if (encrypt\_word[i] == '1')  {  textRange.CharacterFormat.TextColor = Color.Blue;  }  if (encrypt\_word[i] == '0')  {  textRange.CharacterFormat.TextColor = Color.Red;  }  if (encrypt\_word[i] == ' ')  {  textRange.CharacterFormat.TextColor = Color.Aqua;  }  }  if (saveFileDialog1.ShowDialog() == DialogResult.OK)  {  string filename = saveFileDialog1.FileName;  filepath\_document\_1 = Path.GetFullPath(filename);  if (radioButton2.Checked == true)  {  doc.SaveToFile(filepath\_document\_1, FileFormat.Doc);  }  if (radioButton1.Checked)  {  doc.SaveToFile(filepath\_document\_1, FileFormat.Docx);  }  }  } |

Листинг 15.1 – Процесс внедрения скрытой последовательности в документ

Код загружает документ Word по указанному пути, используя document.LoadFromFile(). Затем он извлекает текст из загруженного документа, вызывая document.GetText() и сохраняет его в строковой переменной с именем «text». Затем код проверяет, есть ли в текстовом поле Word какой-либо текст, и если да, он шифрует текст, преобразуя сообщение в код ASCII, а затем в байты, используя методы ConvertMessToByte() и ConvertToASCII().

Затем создается новый экземпляр документа Word с использованием класса Document(), и новый абзац добавляется в документ с помощью метода AddParagraph(). Затем извлеченный текст добавляется к этому новому абзацу путем вызова paragraph1.AppendText().

Затем код вычисляет количество текстовых сегментов, необходимых для шифрования сообщения, и устанавливает цвет каждого сегмента на основе соответствующих им битовых значений. Метод selections.GetAsOneRange() возвращает диапазон выделенного текста в виде отдельного объекта TextRange.

Для извлечения информации из документа используется посимвольный перебор документа и сравнение цвета очередного символа с заданным при зашифровании. Если есть совпадение. Соответствующий символ: 0, 1 или пробел присоединяются к результирующей строке.

Код приведён в листинге 15.2.

|  |
| --- |
| document.LoadFromFile(filepath\_document\_1);  ParagraphCollection paragraph = document.Sections[0].Paragraphs;  foreach (Paragraph par in paragraph)  {  foreach (DocumentObject docObj in par.ChildObjects)  {  if (docObj is TextRange)  {  TextRange textRange = docObj as TextRange;  if (textRange.CharacterFormat.TextColor.Name == "ff0000ff")  {  decryptionMessage += '1';  }  if (textRange.CharacterFormat.TextColor.Name == "ffff0000")  {  decryptionMessage += '0';  }  if (textRange.CharacterFormat.TextColor.Name == "ff00ffff")  {  decryptionMessage += ' ';  }  }  }  }  textBox1.Text = "Извлечённая информации: " + ConvertBiteToMessage(decryptionMessage); |

Листинг 15.2 – Процесс извлечения скрытой последовательности из документа

Второй алгоритм работает похожим образом. В коде, приведённом в листинге 15.3 так же идёт разбиение исходного документа на части для их изменения определённым образом. Но если выше описывался способ разбиения документа со случайным шагом и изменение цвета получившегося в результате разбиения выбранного фрагмента, то здесь разбиение происходит по строкам (абзацам в Microsoft Word) и изменение межстрочного расстояния каждого абзаца.

|  |
| --- |
| int counter = 0;  encrypt\_word = ConvertMessToByte(ConvertToASCII(Word.Text));  Section section = doc.AddSection();  for (int i = 0; i < encrypt\_word.Length; i++)  {  string textBuf = text.Substring(counter, text.IndexOf("\n", counter)-counter-1);  counter = text.IndexOf("\n", counter) + 1;  Paragraph paragraph1 = section.AddParagraph();    if (encrypt\_word[i] == '1')  {  paragraph1.Format.LineSpacing = 20;  }  if (encrypt\_word[i] == '0')  {  paragraph1.Format.LineSpacing = 50;  }  if (encrypt\_word[i] == ' ')  {  paragraph1.Format.LineSpacing = 19;  }  paragraph1.AppendText(textBuf);  }  if (saveFileDialog1.ShowDialog() == DialogResult.OK)  {  string filename = saveFileDialog1.FileName;  filepath\_document\_1 = Path.GetFullPath(filename);  if (radioButton2.Checked == true)  {  doc.SaveToFile(filepath\_document\_1, FileFormat.Doc);  }  if (radioButton1.Checked)  {  doc.SaveToFile(filepath\_document\_1, FileFormat.Docx);  }  } |

Листинг 15.3 – Процесс внедрения скрытой последовательности в документ

Единице исходного сообщения в бинарном представлении соответствует отступ в 20 пунктов, нулю – в 50, пробелу, чтобы было не так заметно, 19 пунктов.

Для извлечения внедрённой последовательности также происходит построчный перебор нового документа и сверение настроек его межстрочного расстояния с объявленными значениями для получения на выходе зашифрованной последовательности.

Код приведён в листинге 15.4.

|  |
| --- |
| document.LoadFromFile(filepath\_document\_1);  ParagraphCollection paragraph = document.Sections[0].Paragraphs;  foreach (Paragraph par in paragraph)  {  if (par.Format.LineSpacing == 20)  {  decryptionMessage += '1';  }  if (par.Format.LineSpacing == 50)  {  decryptionMessage += '0';  }  if (par.Format.LineSpacing == 19)  {  decryptionMessage += ' ';  }  }  textBox1.Text = "Извлечённая информации: " + ConvertBiteToMessage(decryptionMessage); |

Листинг 15.4 – Процесс извлечения скрытой последовательности из документа

Наглядное изменение цвета представленно на рис. 15.1

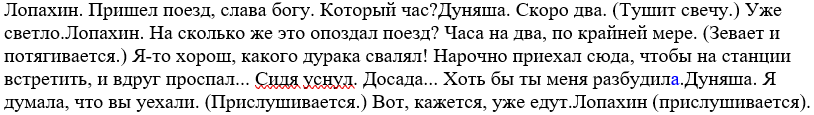


Рис. 15.1 ­– изменение цвета буквы

**Вывод**: исследование было посвящено методам текстовой стеганографии. Были исследованы различные подходы и техники для скрытой передачи информации в текстовых данных. Текстовая стеганография позволяет встраивать секретные сообщения в текстовые файлы, используя различные методы, такие как изменение шрифтов, вставка пробелов или замена символов. Целью исследования было выявить эффективные и незаметные методы стеганографии в текстовом формате. Полученные результаты могут быть полезными для разработки систем защиты данных и обмена конфиденциальной информацией через текстовые каналы связи.