**УДК 004.021**

# РАЗРАБОТКА СИСТЕМЫ ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ НА ОСНОВЕ АЛГОРИТМА ПОСЛЕДОВАТЕЛЬНОЙ СТЕГАНОГРАФИИ

***И.С. АНДРЕЕВ***
***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)***

*Рассматриваются: алгоритм скрытия информации в изображении на основе последовательной стеганографии, описание работы основной программы и основные функции разрабатываемого программного продукта.*

**Введение.** Современный этап в развитии обмена информацией, который характеризуется интенсивным внедрением современных информационных технологий, широким распространением локальных, корпоративных и глобальных сетей во всех сферах жизни цивилизованного государства, создает новые возможности и качество информационного обмена. И поэтому проблемы информационной безопасности постоянно усугубляются процессами проникновения во все сферы общества технических средств обработки и передачи данных и, прежде всего, компьютерных системах. Для решений этих проблем и используют стеганографию, задача которой – передать данные так, чтобы противник не догадался о самом факте появления сообщения.

**Основная часть.** Цель работы – изучение стеганографического метода скрытия информации в изображениях как с точки зрения устойчивости к различным видам атак, так и с точки зрения сохранения приемлемого качества изображения.

Алгоритм последовательной стеганографии заключается в последовательном встраивании одного изображения в другое, что позволяет достичь большего коэффициента сжатия.

Основными функциями разрабатываемого программного продукта являются:

– шифрование информации;
– дешифрование информации;
– вывод промежуточной информации о выполненных операциях.

Рассматриваемая методика ляжет в основу разрабатываемого программного продукта «Скрытие информации на основе дискретных преобразований методом последовательной стеганографии», который реализует все задачи, в частности, главную – скрыть факт передачи информации.

Для скрытия информации методом последовательной стеганографии необходимо выполнить следующие пункты:

1. Изображение-контейнер разбивается на равные блоки.
2. Определяется размер встраиваемого сообщения и количество требуемых битов в контейнере для встраивания.
3. В каждом блоке происходит поиск наименее значащих бит.
4. Создается матрица встраивания (которая определяет в каких битах изображения будет происходить изменения).
5. С помощью алгоритма Блюм-Блюм-Шуба создается последовательность для матрицы встраивания.
6. Происходит замена наименее значащих бит в соответствии с матрицей встраивания.

Извлечение:

1. Изображение-контейнер разбивается на равные блоки.
2. С помощью специального кода из изображения извлекается информация о размере встроенного сообщения.
3. Создается матрица извлечения.
4. Происходит извлечение и сохранение изображения.

Обнаружение кодированного сообщения осуществляется по аномальным характеристикам распределения значений диапазона младших битов отсчётов цифрового сигнала.

Кодирование в программе реализовано следующим образом:

**Листинг 1** – Описание функции, предназначенной для кодирования

```
1. public void bitsToFile(ArrayList<Integer> bits, String filename, String ext, int byteCount) throws
   Exception {
2. System.out.println(bits.size()+" "+ filename+" "+ ext + " "+byteCount);
```

```
3.  byte[] bytes = new byte[byteCount];
4.  int j = 0;
5.  String str = "";
6.  for (int i = bits.size() - byteCount * 8; i < bits.size(); i++) {
7.  str += bits.get(i);
8.  if (str.length() == 8) {
9.  int number = Integer.parseInt(str, 2);
10. str = "";
11. bytes[j] = (byte) number;
12. j++;}
13. }
14. File fl = new File(filename);
15. String path = filename.replaceAll(fl.getName(), "");
16. path += "\\extractedMessage." + ext;
17. FileOutputStream fos = new FileOutputStream(new File(path));
18. fos.write(bytes);
19. fos.close();
20. result = new String[2];
21. result[0] = "true";
22. result[1] = path;
23. }
```

Декодирование в программе реализовано следующим образом:

**Листинг 2** – Описание функции предназначенной для декодирования

```
1.  public void loadMessage(String filename) throws Exception {
2.  FileInputStream fis = new FileInputStream(filename);
3.  byte[] fileBArray = new byte[fis.available()];
4.  fis.read(fileBArray);
5.  fis.close();
6.  String ex = filename.substring(filename.lastIndexOf(".") + 1);
7.  String str = ex + "." + fileBArray.length + "~";
8.  byte[] tmp = str.getBytes();
9.  int[] result = new int[fileBArray.length + tmp.length];
10. int j = 0;
11. for (int i = 0; i < tmp.length; i++) {
12. if (tmp[i] < 0) {
13. result[j] = 256 + tmp[i];
14. } else {
15. result[j] = tmp[i];
16. }
17. j++;
18. }
19. for (int i = 0; i < fileBArray.length; i++) {
20. if (fileBArray[i] < 0) {
21. result[j] = 256 + fileBArray[i];
22. } else {
23. result[j] = fileBArray[i];
24. }
25. j++;
26. }
27. message = result;
28. }
```

Для выбора порядка встраивания сообщения будет использоваться генератор ПСЧ «Алгоритм Блюм — Блюма — Шуба».

**Заключение.** В данной статье был рассмотрен алгоритм скрытия данных в изображении на основе последовательной стеганографии, описание работы основной программы и основные функции разрабатываемого программного продукта.

ЛИТЕРАТУРА

1. Blum, L. A Simple Unpredictable Pseudo-Random Number Generator / Lenore Blum, Manuel Blum, Michael Shub // SIAM Journal on Computing. – 1986. – Vol. 15. – P. 364–383.
2. Садов, В.С. Компьютерная стеганография (конспект лекций) / В.С. Садов. – Минск : Белорус. гос. ун-т, 2010.
3. Конахович, Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю. Пузыренко. – К. : МК-Пресс, 2006. – 288.
4. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И.Н. Оков, И.В. Туринцев. – М. : Солон-Пресс, 2002.