Вистунов С.С., Жиляков Г.В. Изучение работы алгоритма наименьших значимых битов и его реализация на языке PYTHON // Международный журнал информационных технологий и энергоэффективности. – 2024. –
Т. 9 № 9(39) с. 31–38

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.056.55

ИЗУЧЕНИЕ РАБОТЫ АЛГОРИТМА НАИМЕНЬШИХ ЗНАЧИМЫХ БИТОВ И ЕГО РЕАЛИЗАЦИЯ НА ЯЗЫКЕ PYTHON

[1]Вистунов С.С., Жиляков Г.В.
ФГБОУ ВО "САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА", Санкт-Петербург, Россия (193232, г. Санкт-Петербург, пр. Большевиков д.22, корп.1), e-mail:[1] solekvis@yandex.ru

В данной статье представлен результат работы стенографического алгоритма по вложению в наименьшие значимые биты и приведён один из возможных методов реализации алгоритма шифрования НЗБ на языке python.

Ключевые слова: НЗБ, стеганография, шифрования, вложение, программа.

STUDYING THE OPERATION OF THE LEAST SIGNIFICANT BITS ALGORITHM AND ITS IMPLEMENTATION IN PYTHON

[1]Vistunov S.S., Zhilyakov G.V.
BONCH-BRUEVICH ST. PETERSBURG STATE UNIVERSITY OF TELECOMMUNICATIONS, St. Petersburg, *Russia (193232, St. Petersburg, 22 Bolshevikov Ave., bldg. 1), e-mail: [1]solekvis@yandex.ru*

This article presents the result of the least significant bits algorithm and provides one of the possible methods for implementing the LSB encryption algorithm in python.

Keywords: LSB, steganography, encryption, attachment, program.

Введение

Стеганография — это наука о скрытой передаче информации [1]. Она занимается методами и техниками встраивания и извлечения информации в/из носителей данных (например, изображений, звуковых файлов, текстовых документов) таким образом, чтобы сохранить целостность и неизменность файла [2]. Суть стеганографии заключается в том, чтобы скрыть наличие самой информации, т.е. сделать ее незаметной для посторонних наблюдателей [3].

Существует множество алгоритмов вложения информации, однако, в данной статье рассмотрен алгоритм вложения в наименьшие значащие биты (НЗБ) в изображение [4].

Алгоритм вложения в наименьших значимых битов

НЗБ – это один из наиболее распространенных методов стеганографии, который используется для скрытого встраивания информации в носитель данных, такие как изображения, звуковые файлы или видео [5].

В методе НЗБ встраиваемые данные (обычно биты) заменяют младшие (наименее значащие) биты пикселей или сэмплов звукового файла. Поскольку младшие биты обычно содержат меньше информации и меньше влияют на визуальное или аудио восприятие, замена их позволяет скрыть встраиваемую информацию в носитель незаметным образом [6].

Изображение состоит из пикселей, которые в свою очередь и формируют привычные изображения[7]. Сами пиксели получают свой оттенок благодаря сочетанию красного, синего и зелёного цвета, соответственно каждый пиксель представляется в виде матрицы из трёх значений от 0 до 255, что приводит его к виду матрицы (от 0,0,0 до 255,255,255).

Алгоритм вложения НЗБ подразумевает вложение информации в значения красного, зелёного и синего цветов [8]. Это осуществляется за счёт того, что значения, принимаемые каждым из цветов, являются восьмибитными числами, что в свою очередь позволяет представить матрицу значений цветов в восьмибитном формате.

Сам алгоритм выглядит следующим образом.

Например, матрица яркостей пикселей состоит из следующих значений:

[[(225, 12, 99), (155, 2, 50), (99, 51, 15), (15, 55, 22)],

[(155, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66)],

[(219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]]

Если необходимо произвести вложение в изображение слова “hi”, которое по таблице ASCII выглядит как 0110100 0110101, необходимо выбирать значение пикселей один за другим (либо по стегоключу) и заменять их наименее значимые биты в восьмибитном формате на соответствующие биты вкладываемого сообщения [9]. Таким образом, значение 225 из матрицы в восьмибитном формате выглядит как 11100001, а первый бит дополнительного сообщения равен 0. После замены бита получено значение 11100000, что равняется 224. Повторяя данный алгоритм, производится запись сообщения в матрицу до тех пор, пока всё сообщение не будет записано полностью [10].

Результат вложения сообщения в матрицу выглядит следующим образом.

[[(224, 13, 99), (154, 3, 50), (98, 50, 15), (15, 54, 23)],

[(154, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66)],

[(219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]]

Для того чтобы декодировать такое сообщение необходимо произвести алгоритм в обратном порядке [7]. Для того чтобы понять в какой момент декодировки были получены все биты вложенного сообщения необходимо использовать стоп маркеры, которые сигнализируют что сообщение окончено. (Рисунок 2).

Реализация алгоритма на языке python

В представленной реализации используются две библиотеки “Cv2” и “NumPy”.

В начале необходимо конвертировать полученные данные дополнительного изображения в бинарный вид, которая получена с помощью функции представленной на Рисунке 1.

```python
def to_bin(data):
    """Convert `data` to binary format as string"""
    if isinstance(data, str):
        return ''.join([format(ord(i), "08b") for i in data])
    elif isinstance(data, bytes):
        return ''.join([format(i, "08b") for i in data])
    elif isinstance(data, np.ndarray):
        return [format(i, "08b") for i in data]
    elif isinstance(data, int) or isinstance(data, np.uint8):
        return format(data, "08b")
    else:
        raise TypeError("Type not supported.")
```

Рисунок 1 – Конвертация типа данных

После этого прописывается функцию кодирования сообщения в выбранное изображение. В этот момент необходимо прописать команду отвечающую за стоп-маркер (Рисунок 2).

```python
def encode(image_name, secret_data):
    # read the image
    image = cv2.imread(image_name)
    print(image)
    # maximum bytes to encode
    n_bytes = image.shape[0] * image.shape[1] * 3 // 8
    print("[*] Maximum bytes to encode:", n_bytes)
    if len(secret_data) > n_bytes:
        raise ValueError("[!] Insufficient bytes, need bigger image or less data.")
    print("[*] Encoding data...")
    # add stopping criteria
    secret_data += "====="
    data_index = 0
    # convert data to binary
    binary_secret_data = to_bin(secret_data)
    # size of data to hide
    data_len = len(binary_secret_data)
    for row in image:
        for pixel in row:
            # convert RGB values to binary format
            r, g, b = to_bin(pixel)
            # modify the least significant bit only if there is still data to store
            if data_index < data_len:
                # least significant red pixel bit
                pixel[0] = int(r[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                # least significant green pixel bit
                pixel[1] = int(g[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            if data_index < data_len:
                # least significant blue pixel bit
                pixel[2] = int(b[:-1] + binary_secret_data[data_index], 2)
                data_index += 1
            # if data is encoded, just break out of the loop
            if data_index >= data_len:
                break
    print(image)
    return image
```

Рисунок 2 – Кодирование сообщения

Следующим шагом является прописывание кода отвечающего за декодирование сообщения. В данном фрагменте выполняется преобразование полученных данных в биты, после чего осуществляется декодирование до стоп маркера (Рисунок 3.).

```python
def decode(image_name):
    print("[+] Decoding...")
    # read the image
    image = cv2.imread(image_name)
    binary_data = ""
    for row in image:
        for pixel in row:
            r, g, b = to_bin(pixel)
            binary_data += r[-1]
            binary_data += g[-1]
            binary_data += b[-1]
    # split by 8-bits
    all_bytes = [binary_data[i: i + 8] for i in range(0, len(binary_data), 8)]
    # convert from bits to characters
    decoded_data = ""
    for byte in all_bytes:
        decoded_data += chr(int(byte, 2))
        if decoded_data[-5:] == "=====":
            break
    return decoded_data[:-5]
```

Рисунок 3 – Декодирование сообщения

На Рисунке 4 изображен фрагмент кода программы, в котором прописываются элементы для запуска программы, такие как название исходного файла, название зашифрованного файла, секретное сообщение, которое необходимо вложить в исходный файл.

```python
if __name__ == "__main__":
    input_image = "start.jpg"
    output_image = "encoded_image.jpg"
    secret_data = "This is a top secret message."
    # encode the data into the image
    encoded_image = encode(image_name=input_image, secret_data=secret_data)
    # save the output image (encoded image)
    cv2.imwrite(output_image, encoded_image)
    # decode the secret data from the image
    decoded_data = decode(output_image)
    print("[+] Decoded data:", decoded_data)
```

Рисунок 4 – Код вывода данных программы

Далее будет рассмотрен пример работы программы.

На Рисунке 5 представлен фрагмент матрицы исходного файла.

```
[[[207 219 219]
  [207 219 219]
  [207 219 219]
  ...
  [119 138 141]
  [120 139 142]
  [121 140 143]]

 [[210 222 222]
  [210 222 222]
  [210 222 222]
```

Рисунок 5 – Фрагмент матрицы исходного файла

На Рисунке 6. продемонстрировано изменение значений матрицы исходного файла, из чего следует что секретное сообщение успешно вложено.

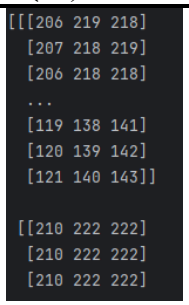

Рисунок 6 – Фрагмент матрицы файла с вложением

На Рисунке 7. продемонстрирован исходный файл без вложения.

Рисунок 7 – Исходный файл

На Рисунке 8. продемонстрирован файл с вложением. В приведённом изображении человеческий глаз не сможет распознать изменённые пиксели.

Рисунок 8 – Файл с вложением

На Рисунке 9. продемонстрирован результат работы функции декодирования.

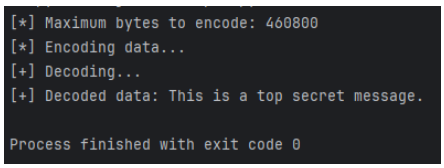

Рисунок 9 – Результат выполнения программы

ЗАКЛЮЧЕНИЕ

В данной статье, приведён метод работы алгоритма НЗБ для вложения сообщений в изображения, а также представлена программа на языке Python для его реализации. Данный алгоритм является простым для реализации и использования для вложения информации в изображения и аудио файлов, который позволяет вложить сообщение незаметное для человеческого восприятия, а также обеспечивает относительно высокую ёмкость для встраиваемой информации. Однако у алгоритма имеется ряд недостатков, таких как уязвимость к стеганализу, что означает возможность обнаружения или извлечения скрытой информации, ограничение форматов данных: алгоритм НЗБ может быть применен только к определенным форматам данных, таким как изображения или звуковые файлы, где есть доступ к отдельным пикселям или сэмплам. Для других форматов данных, таких как текстовые документы или видео, требуются другие методы стеганографии.

Список литературы

1. Цифровая стеганография и водяные цифровые знаки Коржик В.И., Небаева К.А., Герлинг Е.Ю., Догиль П.С., Федянин И.А. Под общей редакцией профессора В.И. Коржика. Санкт-Петербург, 2016. Том Часть 1 Цифровая стеганография
2. Ахрамеева, К. А. Сравнительный анализ стегосистем с вложением в наименьшие значащие биты с согласованием и с замещением / К. А. Ахрамеева, Е. Ю. Герлинг // Наукоемкие технологии в космических исследованиях Земли. – 2020. – Т. 12, № 6. – С. 38-47. – DOI 10.36724/2409-5419-2020-12-6-38-47. – EDN WRCCJX.
3. Ахрамеева, К. А. Автоматизация визуального метода стегоанализа на НЗБ / К. А. Ахрамеева, Е. Ю. Герлинг, В. Е. Радынская // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2020. – № 1. – С. 42-45. – DOI 10.46418/2079-8199_2020_1_7. – EDN OXDOWU.
4. Использование метрики BLEU для оценки естественности текста лингвистических стегосистем / К. А. Ахрамеева, Е. Ю. Герлинг, Д. Ю. Мицковский, С. В. Прудников // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ и управление. – 2020. – № 2. – С. 73-80. – DOI 10.25586/RNU.V9187.20.02.P.073. – EDN EMCSGR.
5. Аккафу Аду, Г. Д. Оценка качества ауйдиофайла после вложения информации в блоки наименьших значащих бит / Г. Д. Аккафу Аду, К. А. Ахрамеева, Е. Ю. Герлинг // Региональная информатика и информационная безопасность : Сборник трудов конференций: Санкт-Петербургской международной конференции и Санкт-Петербургской межрегиональной конференции, Санкт-Петербург, 28–30 ноября 2020

года. Том Выпуск 9. – Санкт-Петербург: Региональная общественная организация "Санкт-Петербургское Общество информатики, вычислительной техники, систем связи и управления", 2020. – С. 240-244. – EDN PYLUWH.

6. Ахрамеева К.А., Герлинг Е.Ю., Ковцур М.М., Галецкая А.В. Стегоанализ объектов полученных с помощью стеганографических программ, распространяемых в сети Интернет // StudNet. 2020. Т. 3. №9. С. 1023-1030. EDN: RNFSVF
7. Герлинг Е. Ю., Ахрамеева К. А. Обзор современного программного обеспечения, использующего методы стеганографии // Экономика и качество систем связи. 2019. № 3 (13). С. 51-58. EDN: KEFWXI
8. Shterenberg S. I., Krasov A. V., Ushakov I. A. Analysis of using equivalent instructions at the hidden embedding of information into the executable files // Journal of Theoretical and Applied Information Technology. 2015. Vol. 80. No. 1. С. 28-34. EDN: UVYEWX
9. Годлевский А. К., Коржик В. И. Стегосистемы повышенной секретности для вложения информации в неподвижные изображения // Сборник научных статей V международной научно-технической и научно-методической конференции "Актуальные проблемы инфотелекоммуникаций в науке и образовании". 2016. С. 320-323. EDN: WZILSF
10. Герлинг Е. Ю. Исследование эффективности методов обнаружения стегосистем, использующих вложение в наименее значащие биты // Информационные системы и технологии. 2011. № 4. С. 137-144. EDN: NUREKZ"

References

1. Digital steganography and watermarks Korzhik V.I., Nekhaeva K.A., Gerling E.Yu., Dogel P.S., Fedyanin I.A. Under the general editorship of Professor V.I. Korzhik. St. Petersburg, 2016. Volume Part 1 Digital Steganography
2. Akhrameeva, K. A. Comparative analysis of stegosystems with an investment in the least significant bits with matching and substitution / K. A. Akhrameeva, E. Yu. Gerling // High–tech technologies in space research of the Earth. - 2020. – Vol. 12, No. 6. – pp. 38-47. – DOI 10.36724/2409-5419-2020-12-6-38-47. – EDN WRCCJX.
3. Akhrameeva, K. A. Automation of the visual method of stegoanalysis on NZB / K. A. Akhrameeva, E. Yu. Gerling, V. E. Radynskaya // Bulletin of the St. Petersburg State University of Technology and Design. Series 1: Natural and Technical Sciences. - 2020. – No. 1. – pp. 42-45. – DOI 10.46418/2079-8199_2020_1_7. – EDN OXDOWU.
4. Using the BLEU metric to assess the naturalness of the text of linguistic stegosystems / K. A. Akhrameeva, E. Y. Gerling, D. Y. Mitskovsky, S. V. Prudnikov // Bulletin of the Russian New University. Series: Complex systems: models, analysis and management. - 2020. – No. 2. – pp. 73-80. – DOI 10.25586/RNU.V9187.20.02.P.073. – EDN EMCSGR.
5. Akkafu Adu, G. D. Evaluation of the quality of an audio file after embedding information in blocks of the least significant bits / G. D. Akkafu Adu, K. A. Akhrameeva, E. Yu. Gerling // Regional informatics and information security : Proceedings of conferences: St. Petersburg International Conference and St. Petersburg Interregional Conference, St. Petersburg, 28-30 November 2020. Volume Issue 9. – St. Petersburg: Regional public organization "St. Petersburg Society of Informatics, Computer Technology, Communication and Management Systems", 2020. – pp. 240-244. – EDN PYLUWH.

6. Akhrameeva K.A., Gerling E.Yu., Kovtsur M.M., Galetskaya A.V. Steganalysis of objects obtained using steganographic programs distributed on the Internet // StudNet. 2020. T. 3. No. 9. C. 1023-1030. EDN: RNFSVF
7. Gerling E. Yu., Akhrameeva K. A. Review of modern software using steganography methods // Economics and quality of communication systems. 2019. No. 3 (13). pp. 51-58. EDN: KEFWXI
8. Shterenberg S. I., Krasov A. V., Ushakov I. A. Analysis of using equivalent instructions at the hidden embedding of information into the executable files // Journal of Theoretical and Applied Information Technology. 2015. Vol. 80. No. 1. pp. 28-34. EDN: UVYEWX
9. Godlevsky A. K., Korzhik V. I. High-security stegosystems for embedding information in still images // Collection of scientific articles of the V International scientific, technical and scientific-methodological conference "Actual problems of infotelecommunications in science and education". 2016. pp. 320-323. EDN: WZILSF
10. Gerling E. Y. Investigation of the effectiveness of methods for detecting stegosystems using embedding in the least significant bits // Information systems and technologies. 2011. No. 4. pp. 137-144. EDN: NUREKZ