УДК 004.056.5

Программная система встраивания информации в изображения с использованием стеганографических и криптографических алгоритмов

А.А. Егоров, А.В. Чернышова
Донецкий национальный технический университет
yegorov0725@yandex.ru, alla@pmi.dgtu.donetsk.ua

Егоров А.А., Чернышова А.В. Программная система встраивания информации в изображения с использованием стеганографических и криптографических алгоритмов. *В статье описаны основные структуры и алгоритмы разработанной программной системы защиты информации с использованием стеганографических и криптографических алгоритмов. Приведен авторский алгоритм внедрения и извлечения данных при работе с изображением. Проанализированы характеристики популярных существующих программных средств.*

Введение

Цель работы: исследование стеганографических и криптографических алгоритмов защиты информации, последующее проектирование и разработка программной системы встраивания информации в изображения с использованием стеганографических и криптографических алгоритмов.

Актуальность работы: защита авторского права в настоящий момент играет большую роль при рассмотрении вопросов информационной безопасности. Механизм цифровых водяных знаков популярен как для защиты различных изображений (фото, логотипы), так и для защиты видео-файлов и аудио-файлов. Предлагаемая программная система может использоваться как средство для защиты авторского права пользователя при работе с файлами в формате .bmp и .png. Предложенный программный продукт можно адаптировать для защиты авторского права в видео-файлах после создания дополнительного модуля работы с форматом .avi [1].

При использовании стеганографических алгоритмов передаваемая информация скрывается в файле. Данные, внутри которых размещаются скрытые сообщения, называются стегоконтейнером, а сами скрытые сообщения – стегосообщениями. Канал передачи стегоконтейнера называется стеганографическим каналом или стегоканалом. К стеганографии относятся множества различных методов: условные расположения знаков, микрофотоснимки, средства связи на плавающих частотах и тайные каналы. Стеганография является отличным дополнением, но никак не заменой криптографии [2].

Использование авторского алгоритма вместе с существующими криптографическими алгоритмами шифрования позволит более надежно защитить встраиваемую в графические изображения информацию и дальнейшую ее передачу по каналам связи.

Анализ существующих систем

На сегодняшний день существует огромное количество открыто распространяемых программных продуктов, использующих стеганографическую технику. Наиболее популярны инструменты размещаются в Github-репозиториях.

В рамках проекта DarkJPEG разработан стеганографический веб-сервис нового поколения, позволяющий скрывать конфиденциальную информацию в виде незаметного шума в JPEG-изображениях, при этом выделить данную информацию можно только зная заданный при кодировании секретный ключ-пароль. Сервис использует стойкие методы стеганографии для сокрытия самого факта сокрытия информации вместе со стойкими методами криптографии для защиты данных, передаваемых по открытым каналам [3].

Основные особенности:

- использование SHA-3 для генерации ключей;
- симметричное шифрование AES-256;
- JPEG (DCT LSB) стеганография;
- поддержка RarJPEG и двойного сокрытия;
- подбор случайного контейнера;
- вычисления без участия сервера (на клиентской стороне);
- гарантия полной конфиденциальности.[3]

OpenStego – это стеганографическое

приложение, которое предоставляет две возможности:

1. Сокрытие данных (можно скрыть любые данные внутри файла-контейнера).

2. Водяные знаки (встраивание невидимой подписи, например в изображении, с целью обнаружения несанкционированного копирования).

Данная программа предоставляет возможность надежного встраивания водяных знаков при незначительном изменении размера или кадрирования изображения [4]. Отличительной особенностью OpenStego является то, что можно расширить ее возможности с помощью своих плагинов, написанных на Java.

LSB-Steganography – стеганографическая программа для скрытия данных в изображениях, реализована в виде скрипта на языке Python, использующая методы наименьшего значащего бита. Модуль LSBSteg основан на OpenCV (библиотека для работы с изображениями). Он использует незначащий бит каждого пикселя и каждого цвета изображения для скрытия данных [5]. Программа не выполняет никаких дополнительных криптографических преобразований для защиты внедренных данных.

Описание структуры разрабатываемой системы

При разработке системы прежде всего необходимо определить требования к программе. На рисунке 1 показана диаграмма использования разрабатываемой системы.

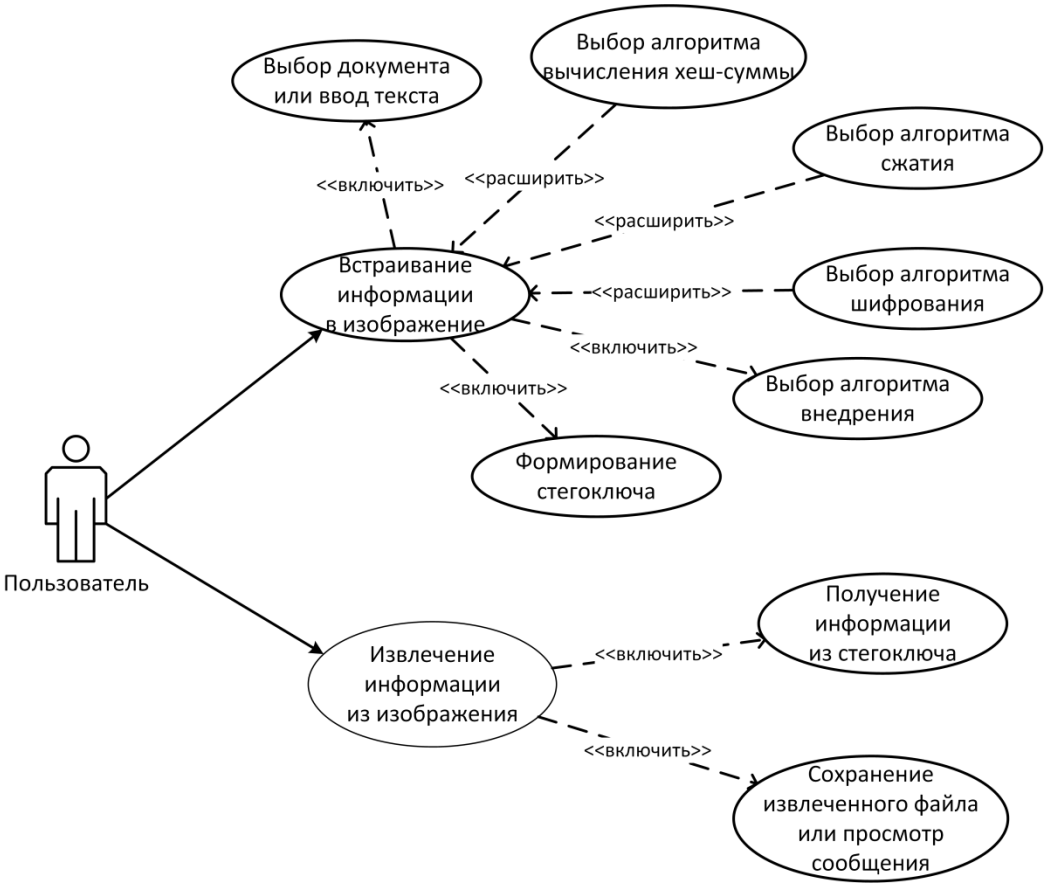

Рисунок 1 – Диаграмма вариантов использования (Uses Case) разрабатываемой системы

При проектировании авторской стегосистемы можно выделить три этапа:

– получение из цифрового документа байтов;
– преобразование байтов цифрового документа (подготовка данных для внедрения в контейнер);
– внедрение данных в изображение, используя метод LSB [6].

На первом этапе некоторый цифровой объект должен быть представлен в виде потока байт. Если это файл, то его можно открыть в бинарном режиме. Если текстовое сообщение, то представить его в виде байтов, заданной кодировкой [7].

На втором этапе, на вход подаются байты, полученные на первом этапе. Прежде всего необходимо вычислить контрольную хеш-сумму [8] исходных байт. После этого сжать

исходные байты, выбранным пользователем алгоритмом сжатия [9]. К сжатым байтам в начало добавляется заголовок (метаданные). После чего эти байты шифруются симметричным алгоритмом шифрования [10].

Последний этап состоит в том, что зашифрованные байты внедряются в изображение.

На рисунке 2 схематически показаны три этапа работы программной системы.

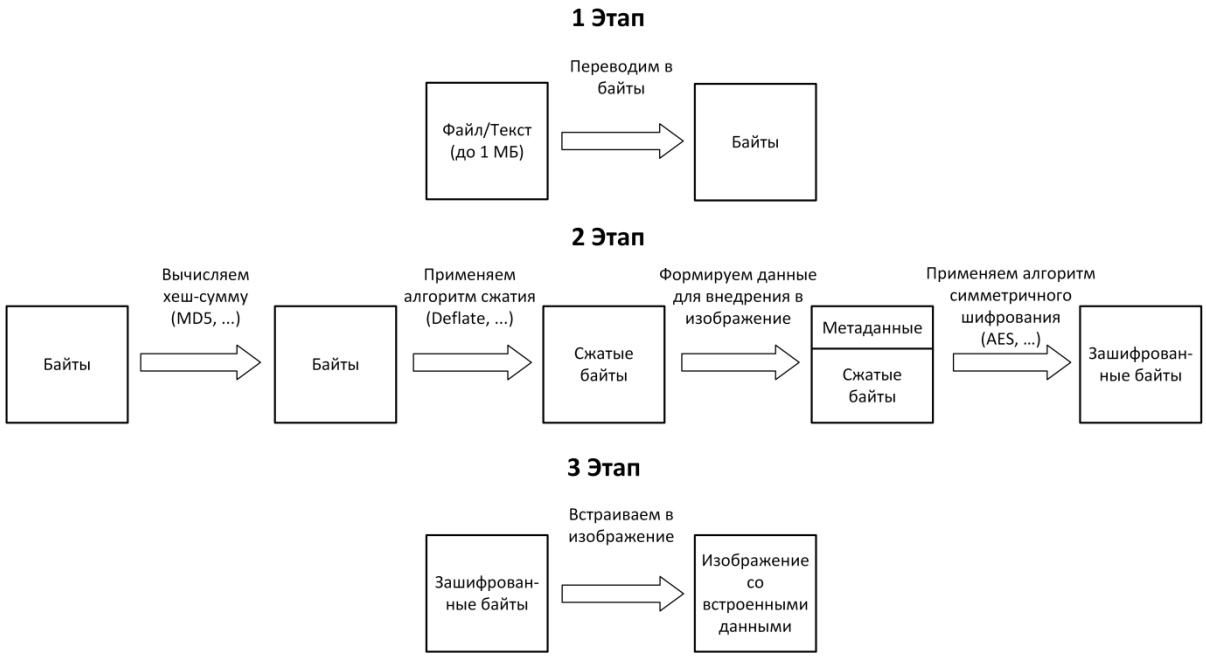

Рисунок 2 – Этапы работы программной системы

Для представления пользовательских данных в виде байтов был разработан интерфейс IData, реализующие этот интерфейс классы должны иметь следующие методы:

– byte[] GetBytes();
– int GetSize();
– Stream GetStream();
– IHeader GetHeader();

В данной системе уже есть два класса, реализующих данный интерфейс: File, Text.

Метаданные – это данные описывающие пользовательские данные. Для представления метаданных был создан интерфейс IHeader. Аналогично реализованы два класса: FileHeader и TextHeader.

UML-диаграмма классов представлена на рисунке 3. Размеры структур и описание полей FileHeader указаны в таблице 1.

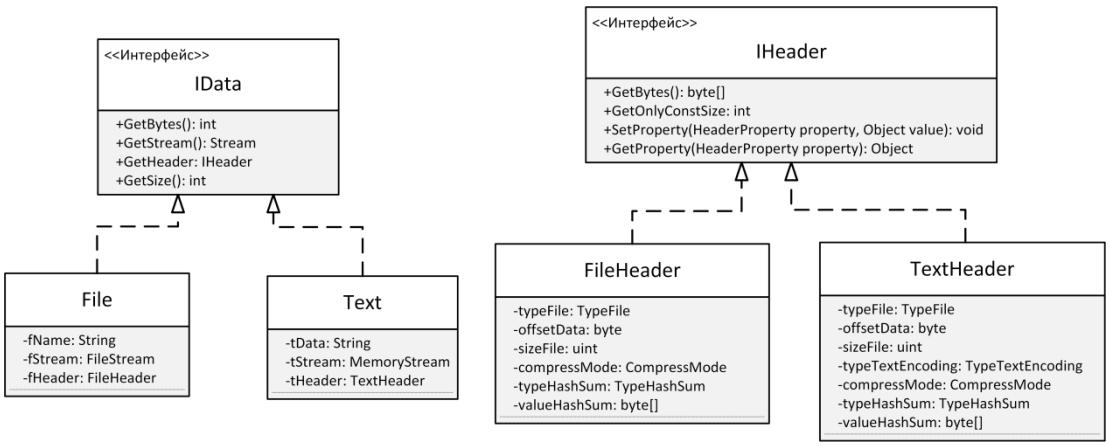

Рисунок 3 – Диаграмма классов, представляющая первый уровень системы

Таблица 1. Структура класса FileHeader

Имя поля	Кол-во байт	Описание
typeFile	1	Тип внедряемого документа
offsetData	1	Смещение начала данных (размер заголовка)
sizeFile	4	Размер файла (кол-во байт)
compressMode	1	Режим сжатия (GZip, Deflate)
typeHashSum	1	Тип хеш-суммы (MD5, SHA-1, …)
valueHashSum	n	Значение хеш-суммы (размер определяется по типу хеш-суммы)

Класс TextHeader имеет такие же поля как FileHeader и дополнительное поле typeTextEncoding, представляющее кодировку внедряемого текста. Данное поле занимает 1 байт (т. е. в системе может быть до 256 различных кодировок текста). В данной системе текст можно представить такими кодировками: ASCII, UTF-8, WINDOWS-1251, CP866.

После представления данных в виде байтов, они проходят ряд преобразований (конвейер трансформации – pipeline transform). Сначала вычисляется контрольная хеш-сумма пользовательских данных, после этого байты сжимаются и шифруются. UML-диаграмма классов второго этапа показана на рисунке 4.

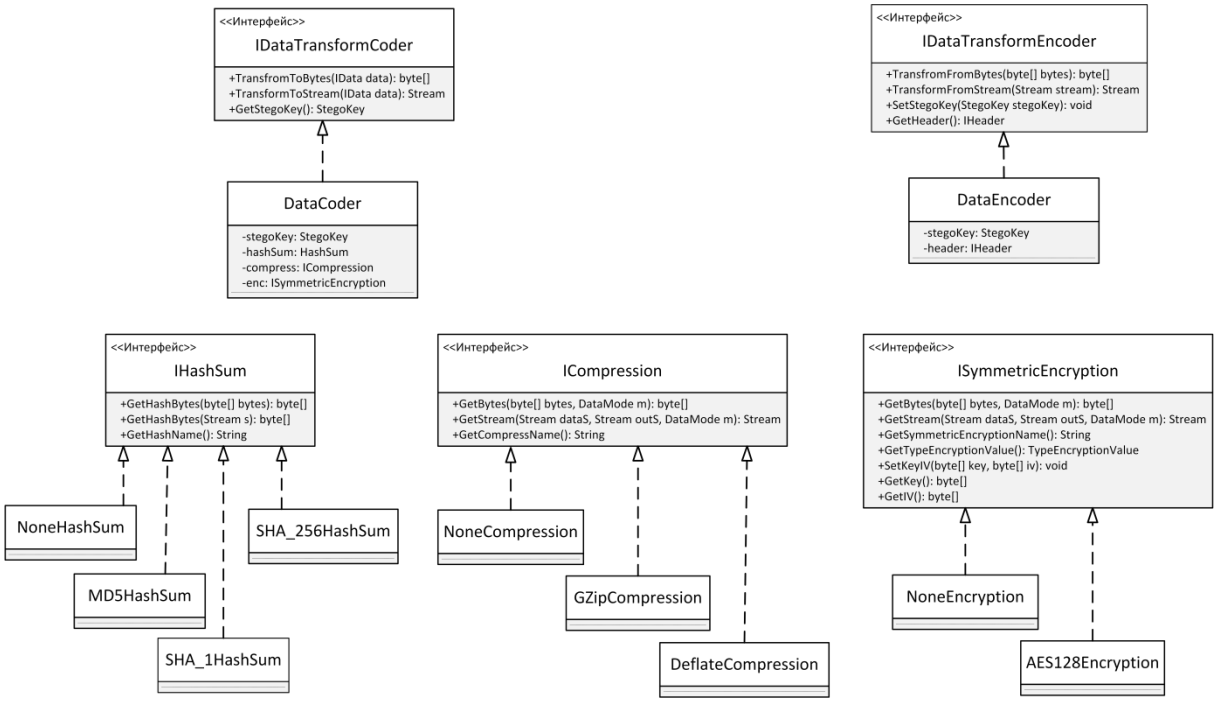

Рисунок 4 – Диаграмма классов второго этапа

Последний этап системы – это внедрение зашифрованных байтов в изображение.

В данной системе создано два интерфейса IStegoTransformEmbed (для внедрения данных в изображение) и IStegoTransformExtract (для извлечения данных из изображения).

Интерфейс IStegoTransformEmbed имеет два метода:

– EmbedFromBytes (внедрение из байтов);
– EmbedFromStream (внедрение из потока);

Метод EmbedFromBytes принимает следующие аргументы:

– int offset – первое смещение начала чтения первого пикселя;
– int maxDiffusion – максимальное значение рассеивания (до 250);
– byte[] bytes – данные для внедрения;
– Bitmap container – изображение, в которое будут внедрятся данные;
– StegoKey key – стегоключ (см таб. 2).

UML-диаграмма классов заключительного этапа показана на рисунке 5. На рисунке 6 показана схема внедрения данных в изображение. На рисунках 7-8 показаны алгоритмы

внедрения/извлечения данных.

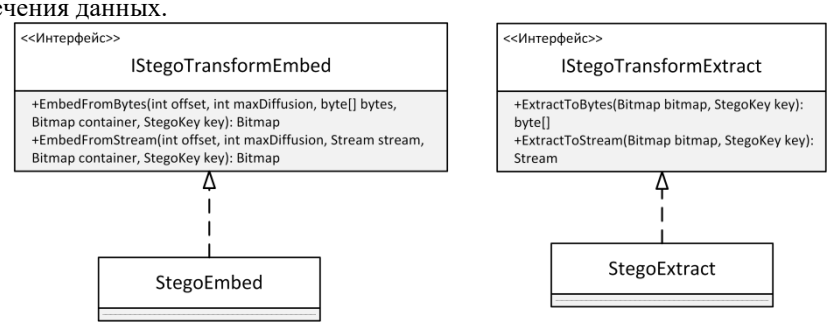

Рисунок 5 – Диаграмма классов заключительного этапа

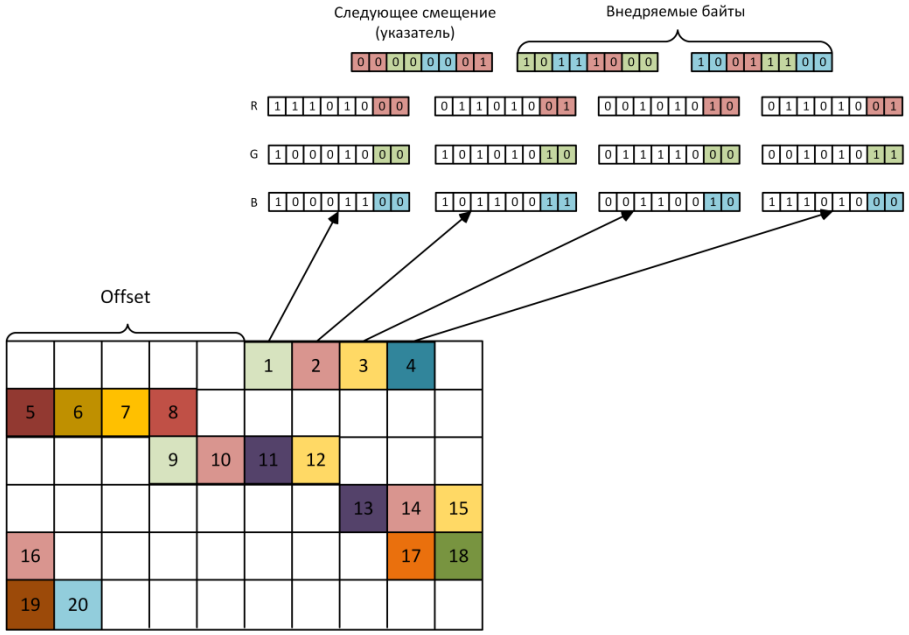

Рисунок 6 – Схема внедрения данных (9-10 байт) в растровое изображения используя метод LSB.

```
Алгоритм ВстраиванияИнформации
  Вход:  offset – первое смещение начала чтения первого пикселя
         maxDiffusion – максимальное значение рассеивания
         bytes – данные для внедрения
         container – растровое изображение для внедрения данных
         key – стегоключ
  Выход: result – изображение со встроенными данными
     Перевести изображение в поток пикселей (по строкам).
     Заполнить стегоключ.
     Пропустить заданное количество offset пикселей.
     Если кол-во внедряемых байт (реальныйРазмер) – четное
     То размерБайтов = реальныйРазмер – 2
     Иначе размерБайтов = реальныйРазмер – 1
     Пока (текущаяПара < размерБайтов)
         Считать из потока 4 пикселя.
         Сгенерировать случайное число – указатель на
         следующие 4 пикселя.
         Заменить последние два бита 4-х пикселей на указатель и
         два байта пользовательских данных.
         Увеличить текущаяПара на два.
     Если кол-во внедряемых байт (реальныйРазмер) – четное
     То Заменить последние два бита 4-х пикселей на 254
        и два байта заключающих данных.
     Иначе (кол-во внедряемых байт - нечетное)
        Заменить последние два бита 4-х пикселей на 255
        и один байт заключающих данных.
     Сформировать изображение из потока пикселей.
     Вернуть сформированное изображение.
```

Рисунок 7 – Алгоритм встраивания информации

```
Алгоритм ИзвлеченияИнформации
  Вход:  container – изображение с внедренными данными
         key – стегоключ
  Выход: байты пользовательских данных
     Перевести изображение в поток пикселей (по строкам).
     Считать информацию из стегоключа.
     Пропустить заданное количество offset пикселей.
     Пока (текущаяПозицияПотока < длинаПотока - 4)
         Считать из потока 4 пикселя.
         Преобразовать последние 2 бита считанных пикселей
         в первый байт – указатель и два пользовательских данных.
         Если указатель равен 255 или 254,
         То Выйти из цикла
         Записать два байта в поток.
     Если указатель равен 255
     То Записать один байт в поток.
     Иначе Если указатель равен 254
     То Записать два байта в поток.
     Вернуть байты из потока.
```

Рисунок 8 – Алгоритм извлечения информации

Таблица 2. Структура класса StegoKey

Имя поля	Кол-во байт	Описание
Offset	4	Первое смещение начала чтения (кол-во пикселей от начала)
EmbededMode	1	Режим встраивания данных в изображение
TypeEncryption	1	Тип симметричного алгоритма шифрования
SizeEncryption-IV	2	Размер вектора инициализации
SizeEncryption-Key	2	Размер ключа шифрования
IV	Указано в SizeEncryptionIV	Вектор инициализации
Key	Указано в SizeEncryptionKey	Ключ шифрования

Из преимуществ стоит отметить, что данная система является расширяемой, например, можно добавить свой алгоритм сжатия или шифрования. Из недостатков – система не является быстродействующей.

Выводы

В статье проведен анализ существующих стегосистем, представлена структура разработанной программной системы, диаграммы классов, некоторые алгоритмы работы программной системы.

Система предоставляет возможность выбирать доступные алгоритмы обработки, которые затем будут записаны в стегоключ. Программную систему можно использовать для различных пользователей. При этом для каждого пользователя настройки программы будут собственными (используемый алгоритм сжатия, используемая кодировка, симметричный алгоритм шифрования, однонаправленная хеш-функция), каждый будет работать со своим стегоключем.

Литература

1. Использование стеганографических и криптографических средств для защиты видеофайлов. А.В. Чернишова, Б.С. Маркин, Наукові праці ДонНТУ Серія “Інформатика, кібернетика та обчислювальна техніка”, №1(19), 2014, стр.46-49
2. Стеганография – актуальность в XXI веке // SeoCyber [Электронный ресурс]. – Режим

доступа: http://seocyber.net/steganografiya-aktualnost-v-xxi-veke
3. DarkJPEG: стеганография для всех // Хабрахабр. [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/187402/
4. OpenStego // Github [Электронный ресурс]. – Режим доступа: https://github.com/syvaidya/openstego
5. LSB-Steganography // Github [Электронный ресурс]. – Режим доступа: https://github.com/RobinDavid/LSB-Steganography
6. LSB стеганография // Хабрахабр. [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/112976/
7. Краткая история кодировок от ASCII до UTF-8 // Popel-studio [Электронный ресурс]. – Режим доступа: http://popel-studio.com/blog/article/kratkaya-istoriya-kodirovok-ot-ascii-do-utf-8.html
8. Контрольные суммы MD5 и SHA. // Desktoplinux [Электронный ресурс]. – Режим доступа: http://desktoplinux.ru/unix_guide/proverka_khesh_summy_md5_v_linux_md5sum
9. Алгоритмы сжатия данных без потерь // Хабрахабр [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/231177/
10. Обзор алгоритмов Шифрования // Rohos [Электронный ресурс]. – Режим доступа: http://www.rohos.com/help/crypto_algorithms.htm

Егоров А.А., Чернышова А.В. Программная система встраивания информации в изображения с использованием стеганографических и криптографических алгоритмов. *В статье описаны основные структуры и алгоритмы разработанной программной системы защиты информации с использованием стеганографических и криптографических алгоритмов. Приведен авторский алгоритм внедрения и извлечения данных при работе с изображением. Проанализированы характеристики популярных существующих программных средств.*

Ключевые слова: *стеганография, ключ стего, LSB метод, криптография, MD5, AES, сжатие*

Yegorov A., Chernyshova A. Software system for embedding data inside images using steganographic and cryptographic algorithms.*The article describes the main structures and algorithms of the developed information security software system that uses steganographic and cryptographic algorithms. The author's algorithm of data embedding and extraction for images is shown. We have analyzed the characteristics of the existing popular software tools.*

Key words:*Steganography, stego key, LSB method, cryptography, MD5, AES, compression*

Статья поступила в редакцию 21.05.2016
Рекомендована к публикации д-ром техн. наук А.С. Миненко