УДК 004.056

Стегоанализ графических данных на основе методов сжатия

М. Ю. Жилкин

На примере форматов BMP и JPEG предлагается метод определения наличия скрытых данных, размещённых путём изменения младших битов в графических файлах. Экспериментально показана высокая эффективность алгоритма.

1. Введение

С давних времён и до настоящего времени не теряет своей актуальности задача защиты информации от несанкционированного доступа. Особенно остро проблема стоит в современном обществе, которое часто называют информационным.

Традиционно существовали два направления решения этой задачи: криптография и стеганография. Криптография скрывает смысл сообщения путём его шифрования. Стеганография же, напротив, скрывает сам факт передачи сообщения, маскируя его в другом сообщении, которое выглядит безобидно и не вызывает подозрение у потенциального перехватчика. Наука стеганография имеют богатую историю и широкий спектр различных методов, каждый из которых характерен для своей эпохи. Так, например, в древности применяли тайнопись на табличках, покрытых воском, передавали сообщение в виде татуировки на голове гонца. В начале XX века использовались симпатические чернила, во времена Второй мировой войны распространение получил метод микрофотографий.

Настоящее время – век информационных технологий – диктует свои законы. Бурное развитие вычислительной техники привело к возникновению особой науки, так называемой цифровой компьютерной стеганографии. Появились новые стеганографические методы, в основе которых лежат особенности представления информации в компьютерных файлах, вычислительных сетях и т.п. [1 – 4].

Методы современной компьютерной стеганографии находят применение в области военной и правительственной связи, защиты авторских прав, для решения задач обеспечения информационной безопасности. Актуальность проблемы информационной безопасности постоянно растёт и стимулирует разработку новых методов стеганографии и атак на уже существующие методы.

Повсеместное распространение компьютерной техники и глобальных компьютерных сетей, простота в эксплуатации оборудования и доступность для пользователя стеганографического программного обеспечения позволяют сегодня каждому желающему использовать методы стеганографии при передаче информации. Стоит заметить, что этими методами с лёгкостью могут воспользоваться и злоумышленники, например, для передачи скрыто конфиденциальной информации, коммерческих и государственных секретов и т.п. Поэтому на сегодняшний день остро стоит проблема построения методов обнаружения скрытых данных в передаваемых сообщениях – задача так называемого стегоанализа.

Цель работы – построение эффективного метода «автоматического», т.е. без участия человека, определения факта наличия скрытых данных в различных графических форматах (на примере BMP и JPEG). В данной работе развивается метод, основанный на применении сжатия данных. Среди достоинств метода – высокая скорость, эффективность, применимость к широкому классу методов замены младших битов данных.

В настоящее время цифровая форма информации стала стандартом де-факто. В таком виде данные могут быть легко созданы, подвергнуты различной обработке, переданы без ошибок на дальние расстояния по высокоскоростным каналам передачи. С развитием компьютерной индустрии и рынка программного обеспечения неизбежно возникают новые и новые алгоритмы обработки информации, а вместе с ними и новые форматы хранения данных. Такие традиционные аналоговые виды информации, как текст, звук, изображение, видео и т.д. в цифровом виде могут быть представлены в самых разнообразных форматах, примеры лишь немногих: PDF, DOC, WAV, MP3, OGG, AVI, MPEG, BMP, GIF, JPEG и др. Как правило, эти форматы имеют достаточно сложную внутреннюю структуру.

Во многих стегосистемах скрытая информация записывается путём изменения младших битов данных. Цифровые изображения, звуки, видео и т.д. для восприятия человеком переводятся в аналоговый вид, кроме того, они могут иметь изначально аналоговую природу, поэтому изменения младших знаков в большинстве случаев слабо заметны или вообще незаметны органами чувств человека из-за небольшой доли вносимых искажений. Современное стеганографическое программное обеспечение в большинстве своём использует именно этот подход для «встраивания» скрытой информации в контейнеры [1 – 4]. Контейнером называется файл (или данные), в которые «прячется» (встраивается) скрытая информация.

Идея метода обнаружения факта встраивания скрытой информации основана на том, что включаемые данные статистически независимы от контейнера. Иными словами,

Статья поступила в редакцию 17.06.2008.

существующие статистические закономерности внутри контейнера будут нарушены при добавлении в него «инородных» данных.

Наш подход к проверке статистической независимости информации основан на применении алгоритмов сжатия информации [5]. При добавлении скрытых данных в контейнер, размер при его сжатии вырастает по сравнению с размером при сжатии исходного «пустого» контейнера. Широко распространённые программы-архиваторы легко могут быть использованы для сжатия.

Разработанный метод имеет высокую скорость, позволяет эффективно обнаруживать скрытые данные в автоматическом режиме без необходимости наблюдения, корректировки и другого вмешательства со стороны человека.

2. Основные методы добавления скрытой информации

Независимо от того, какой формат контейнера используется для скрытия информации, изменяемые исходные данные удобно рассматривать в виде последовательности таких байтов, младшие биты которых можно изменять для добавления скрытой информации. Любое стеганографическое программное обеспечение для непосредственного добавления информации переходит именно к такому представлению данных, предварительно выполнив ряд специфичных для формата контейнера преобразований.

Рассматривая данные контейнера как последовательность изменяемых байтов, можно выделить несколько различных методов занесения скрытой информации:

1. *Последовательное заполнение:* информация включается последовательно в каждый младший бит каждого байта последовательности. Если размер включаемых скрытых данных не превышает ёмкость последовательности, то его остаток будет незаполненным. В некоторых случаях к такому методу заполнения контейнера можно применить визуальную атаку (см. рис. 1). Кроме того, незаполненный «хвост» легко замечают компьютерные алгоритмы стегоанализа.

2. *Последовательное заполнение и «дозаполнение»*: модификация предыдущего метода с целью скрыть остаток, который заполняется псевдослучайной последовательностью битов. Метод в целом лучше предыдущего, но обладает одним существенным недостатком: контейнер всегда заполнен на 100%, даже если внести всего один бит скрытых данных. Это легко позволяет обнаружить включение, поскольку величина вносимых искажений постоянна и всегда велика.
3. *«Размытое» (или «разбросанное») заполнение:* те байты последовательности, куда будет занесена информация, выбираются псевдослучайно. Это самый трудоёмкий для стегоанализа метод. В работе результаты приводятся для этого метода заполнения.

Три вышеописанных метода занесения скрытой информации в контейнер являются универсальными. Кроме того, существуют несколько дополнительных, специфичных для формата JPEG методов. Здесь обращается внимание на структуру последовательности и информацию, которая в ней содержится:

4. *Последовательное заполнение в ненулевых байтах*: этот метод аналогичен обычному последовательному заполнению с той особенностью, что нулевые байты исходного контейнера не используются для занесения в них скрытой информации. Это связано с наличием в структуре JPEG очень большого количества нулевых байтов, изменение которых может вызвать подозрение.
5. *Произвольное «разбросанное» заполнение JPEG*: для заполнения используются любые байты последовательности (нулевые и ненулевые), но выбираются в «разбросанном» стиле, кроме того, информация добавляется не во все байты исходных данных. Обычно ёмкость контейнера, т.е. то количество скрытой информации, которое он способен вместить, при таком методе заполнения существенно снижается. Однако это компенсируется высокой сложностью обнаружения такого включения. Большинство современных стеганографических программ, работающих с JPEG, используют этот метод включения. Наш алгоритм стегоанализа способен успешно обнаруживать данные, скрытые этим методом.

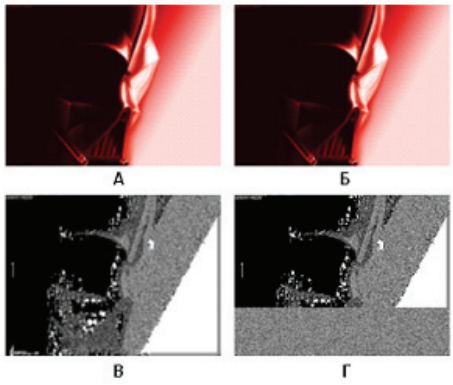

Рис. 1. Пример последовательного заполнения и «Визуальной атаки». Исходные контейнеры: пустой (А), содержащий данные (Б). Младшие биты контейнеров: пустого контейнера (В), непустого контейнера (Г).

3. Краткое описание форматов BMP и JPEG

Аббревиатура BMP означает BitMap («битовая карта»). Как и следует из названия, BMP относится к числу тех форматов, где данные представлены «как есть», без каких-либо преобразований, сжатия и т.д. [6]. По этой причине размер файлов BMP довольно большой, однако формат файла достаточно прост, что делает его очень популярным для применения в качестве стегоконтейнера.

Существуют два способа хранения несжатых графических данных:

- *Индексированный.* Подразумевает наличие так называемой палитры – специального массива данных с описанием всех встречающихся в изображении цветов. Матрица самого изображения содержит лишь индексы палитры. Из всего изображения для включения данных может использоваться лишь палитра, иначе возникнут серьёзнейшие искажения, очень заметные для человека. Однако палитра имеет постоянный размер 1Кб, поэтому позволяет сохранять лишь 128 байт данных независимо от разрешения и размера самого изображения. Существуют также способы включения данных во всё индексированное изображение, однако для этого выполняются преобразования, приводящие к серьёзной потере качества изображения. По этим причинам

индексированные изображения редко используются стеганографическим программным обеспечением. Наш метод стегоанализа работает только с неиндексированными форматами.

- *Неиндексированный.* Этот способ представления данных в основном предназначен для полноцветной графики и широко распространён. Ярким представителем неиндексированных форматов является 24-битный BMP. Он наиболее популярен в среде различных стеганографичеких программ.

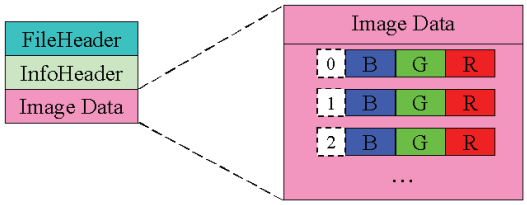

Рис. 2. Структура 24-битного BMP файла.

На рис. 2 представлен формат 24-битного BMP-файла. В самом начале файла располагаются два заголовка с необходимыми служебными полями, за ними следует поле данных, где каждая точка изображения кодируется тремя цветовыми компонентами: синей, зелёной и красной (R, G и B). Под каждую компоненту отводится по одному байту, следовательно, один пиксель изображения занимает 3 байта или 24 бита. При занесении скрытой информации в каждый младший бит максимальная ёмкость контейнера составит примерно 12.5% от его размера.

Формат JPEG разработан в середине 1980-х Объединённой группой экспертов по фотографии (Joint Photographic Expert Group) [7] и отсюда берёт своё название. Целью группы было создание эффективного алгоритма сжатия полноцветных изображений. Этот алгоритм является достаточно сложным, поэтому принято различать:

- «формат JPEG-файла» – способ представления данных JPEG в файле.
- «алгоритм сжатия JPEG» – набор преобразований, переводящих несжатую графическую информацию в сжатые данные JPEG.

Формат JPEG файла в отличие от алгоритма сжатия не был стандартизован, что в недалёком прошлом приводило к проблемам совместимости различного программного обеспечения. В настоящее время общепринятым стандартом считается версия формата, получившая название JFIF. Ввиду определённой запутанности формата файла и наличия в нём всевозможных «фирменных» расширений (например, от Adobe), лишь небольшая часть программ реализуют обработку JPEG-файла самостоятельно. Принятой практикой в настоящее время является использование библиотек поддержки форматов, например IJD JPEG Library[8] для формата JPEG. Все операции по обработке файла и выполнению алгоритма JPEG возлагаются на саму библиотеку. По этой причине мы будем рассматривать только алгоритм сжатия JPEG.

На рис. 3 показаны основные этапы алгоритма JPEG:

1. *Преобразование в цветовое пространство YCbCr.* Исходные данные могут быть в любом цветовом пространстве, например RGB, CMYK и др., а алгоритм JPEG работает с данными в пространстве YCbCr (Y – яркость, Cb и Cr – хроматический синий и хроматический красный соответственно). Изменение компоненты Y гораздо более заметно для человека, чем изменение Cb и Cr. Поэтому сразу после 1-го этапа из матриц Cb и Cr, как правило, отбрасывается каждая вторая строка и каждый второй столбец.
2. *Дискретное косинусное преобразование (ДКП).* Представляет собой двумерное преобразование Фурье. Его основной смысл – перевод данных из пространственного представления в частотное.
3. *Квантование.* Изменение высокочастотных составляющих изображения гораздо менее заметно для глаз, чем изменение низкочастотных. Это даёт возможность заменить все высокочастотные коэффициенты ДКП на нулевые байты. Квантование представляет собой деление коэффициентов ДКП на элементы специальной матрицы квантования с последующим округлением. Это приводит к обнулению большинства высокочастотных коэффициентов.
4. *Сжатие без потерь.* Полученные после третьего этапа данные содержат много повторяющихся нулевых байтов, поэтому эффективно сжимаются с помощью алгоритма кодирования длин повторов (RLE) и затем – алгоритмом Хаффмана.

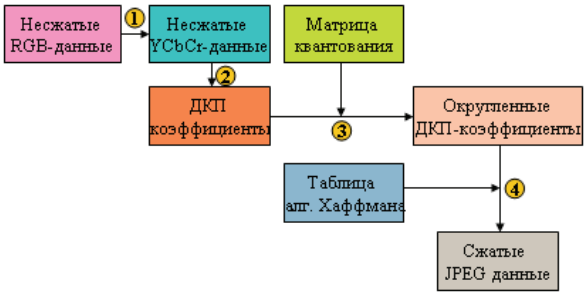

Рис. 3. Алгоритм JPEG. Номерами показаны этапы.

4. Методы включения данных в контейнеры BMP и JPEG

Файл каждого цифрового формата данных имеет свою внутреннюю структуру. Включение скрытой информации подразумевает изменение этой структуры. При включении посторонних данных в контейнер должны выполняться несколько важных условий:

1. При включении данных контейнер не должен быть повреждён, т.е. необходимо сохранить корректную внутреннюю структуру.
2. Изменения в контейнере не должны восприниматься органами чувств человека.
3. Изменения в контейнере должны быть по возможности замаскированы от компьютерных алгоритмов стегоанализа.

Подавляющее большинство стеганографических программ старается удовлетворять по крайней мере первым двум условиям. Кроме того, приведённые выше правила означают, что для каждого формата файла существует свой собственный уникальный алгоритм включения данных. Рассмотрим алгоритмы включения скрытой информации в файлы 24-битного BMP и JPEG.

Поскольку в 24-битном BMP не используется каких-либо алгоритмов сжатия и данные располагаются в чистом виде, то включение скрытой информации можно произвести непосредственно в область данных без дополнитель-

ных преобразований. Важно лишь сохранить заголовки файла неизменными (см. рис. 2).

Алгоритм сжатия JPEG, напротив, включает в себя сложные преобразования исходных данных, поэтому изменять структуры JPEG-файла напрямую нельзя. Кроме того, как известно, JPEG сжимает изображения с потерями, значит, добавление скрытой информации в исходные данные до начала всех преобразований приведёт к разрушению скрытых данных алгоритмом JPEG. Поэтому здесь применяется включение посторонних данных в коэффициенты дискретного косинусного преобразования, которые остаются неизменными (см. рис. 4).

При этом возможны два варианта встраивания:

1. Создание файла JPEG «с нуля» из другого формата с добавлением скрытой информации на предпоследнем этапе (см. рис. 3, 4)
2. Встраивание скрытых данных в готовый JPEG-файл. Сначала необходимо получить ДКП-коэффициенты путём раскодирования данных этапа №4, затем добавить скрытое сообщение и опять выполнить этап №4. (см. рис. 3, 4)

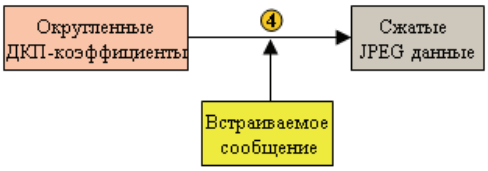

Рис. 4. Способ встраивания данных в JPEG

5. Описание разработанного алгоритма анализа

Перейдём к формальному описанию алгоритма.

Пусть $X = \{x_1, \ldots, x_N\}$ – последовательность байтов в данных изображения. Для формата BMP рассматривается само поле данных, в JPEG – матрица округлённых ДКП-коэффициентов. Пусть $|X| = N$ – длина последовательности. Разобьём последовательность X на d равных отрезков и обозначим каждый отрезок X_i, где $i = 1, 2, \ldots, d$. Пусть $\psi(X)$ – алгоритм сжатия, применённый к последовательности .

Тогда обозначим

$$f(X, n) = \frac{|\psi(X_n)|}{|X_n|}$$

коэффициент сжатия отрезка n последовательности X алгоритмом ψ. Обозначим $\varphi(X)$ псевдослучайное изменение младших битов всех байтов последовательности X.

Пусть X – последовательность, которая подаётся на вход программе, а $Y = \varphi(X)$ – полученная из неё новая последовательность. Исходная последовательность X сжимается сильнее по сравнению с изменённой последовательностью Y.

Введём новую величину

$$(X, n) = |f(X, n) - f(Y, n)|.$$

Те отрезки последовательности X, которые не содержали «скрытую» информацию, сжимаются лучше, чем соответствующие им отрезки последовательности Y, и напротив, коэффициенты сжатия отрезка последовательности X со «спрятанной» информацией и отвечающего ему отрезка последовательности Y отличаются незначительно. Для определения факта включения информации экспериментально подбирается пороговое значение для величины δ и производится оценка количества отрезков, на которых значение величины не превышает порог.

6. Экспериментальный анализ

Для экспериментального исследования метода была подготовлена серия изображений («контейнеров») форматов 24-bit BMP и JPEG разного разрешения и качественного содержания.

Обработка одного изображения выполнялась по следующему алгоритму:

- Вход: пустой контейнер, имя архиватора для выполнения сжатия, пороговое значение δ.
- Тестирование контейнера разработанным алгоритмом анализа со сжатием заданным архиватором.
- Определение факта заполнения по заданному значению δ.
- Вывод результата: «Заполнен» или «Не заполнен»

Ситуация, когда тест отвечает «Заполнен» на пустом контейнере, называется *ошибкой I рода*. *Ошибка II рода* возникает в случае ответа «Не заполнен» при проверке непустого контейнера.

Тестирование большой серии изображений проводилось в несколько этапов:

1. *Настройка.* На этом этапе анализировалась небольшая выборка (около 50 изображений) и подбирался наиболее подходящий архиватор и два пороговых значения следующим образом:

   a. $\psi(X)$ и $\delta = \delta_{\min}$, обеспечивающие отсутствие ошибок I рода.

   b. $\psi(X)$ и $\delta = \delta_{\max}$, при которых достигается «золотая середина» – минимальный процент ошибок II рода при небольшом количестве ошибок I рода.
2. *Тестирование по независимым данным.* После экспериментального подбора архиватора и двух вариантов порогового значения δ проводилась проверка результатов на серии из 1000 других изображений.

Наиболее подходящим архиватором оказался ZIP для формата BMP и встроенный алгоритм (RLE + Хаффман) для JPEG. Были выбраны пороговые значения: $\delta_{\min} = 0,8\%$ и $\delta_{\max} = 1,4\%$

На рис. 5,6 приведены данные по работе алгоритма с архиватором ZIP для BMP и встроенным алгоритмом сжатия для JPEG.

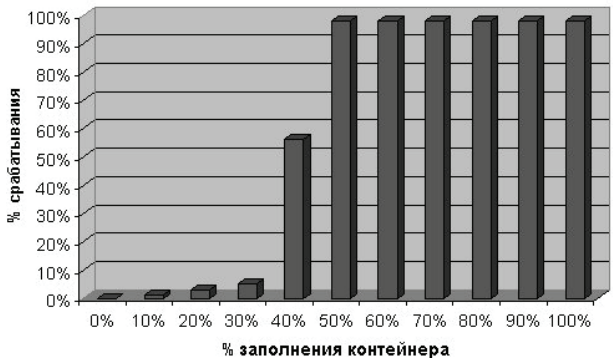

Рис. 5. Результат работы алгоритма на контейнерах формата BMP с независимыми данными при

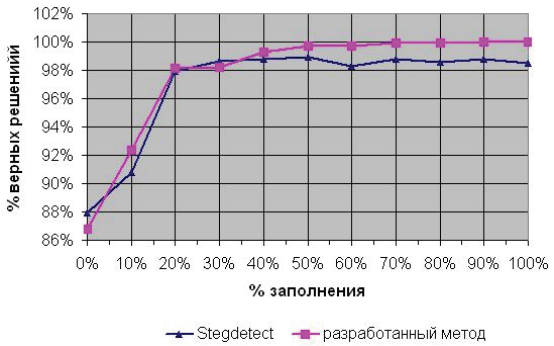

Рис. 6. Результат работы алгоритма на контейнерах формата JPEG в сравнении со стандартной утилитой StegDetect.

7. Выводы

Таким образом, предлагаемый метод позволяет надёжно без участия человека выявлять факт включения данных в изображение. Ошибка при заполнении контейнера на 40% и более не превышает 2%.

Литература

1. A. Westfeld, A. Pfitzmann. Attacks on Steganographic Systems. Breaking the Steganographic Utilities EzStego, Jsteg, Steganos and S-Tools – and Some Lessons Learned. Lecture Notes in Computer Science, 1768:61–75, 2000.
2. N. Provos, P. Honeyman. Hide and Seek: An Introduction to Steganography. IEEE Security & Privacy, May/June 2003, pp. 32–44.
3. O. Dabeer, K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath. Detection of hiding in the least significant bit. In IEEE Trans. on Signal Processing, volume 52, pages 3046–3058, Oct. 2004.
4. J. Fridrich, M. Goljan, and R. Du. Reliable Detection of LSB Steganography in Color and Grayscale Images. Proc. of the ACM Workshop on Multimedia and Security, pp. 27–30, 2001.
5. B. Ryabko, J. Astola. Universal Codes as a Basis for Time Series Testing. Statistical Methodology, v.3, pp.375–397, 2006.
6. BMP format in MSDN – <http://msdn.microsoft.com/en-us/library/ms532311.aspx>
7. JPEG committee – <http://www.jpeg.org>
8. Independent JPEG Group library for JPEG image compression - http://www.ijg.org

Жилкин Михаил Юрьевич

аспирант, ст. преп. кафедры прикладной математики и кибернетики СибГУТИ, инженер лаборатории защиты информации СибГУТИ,
тел. (383) 2-698-272, e-mail: myz@csu.ru