# Построение стегосистемы на базе растровых изображений методом перестановок*

Е. Ю. Елтышева[1], А. Н. Фионов[1,2]
[1] *Сибирский государственный университет телекоммуникаций и информатики*
[2] *Институт вычислительных технологий СО РАН, Новосибирск, Россия*
e-mail: a.fionov@ieee.org, Katerina.Artist@yandex.ru

Предлагается метод сокрытия информации в пикселях растрового изображения, записанного в файлах формата BMP, PNG и др., использующих неискажающие методы сжатия. В отличие от известных аналогов данный метод внедряет сообщение в растр путем незначительных перестановок соседних байт яркостей. Проводится RS-анализ разработанного алгоритма и сравнение его с известными аналогами, демонстрирующее преимущество предложенного алгоритма.

*Ключевые слова*: стеганография, стегоанализ, совершенные стегосистемы, LSB-методы, арифметическое кодирование.

## Введение

Цель стеганографии состоит в организации передачи секретных данных таким образом, чтобы сам факт передачи был скрыт от всех незаконных получателей. Такую задачу можно решить, встроив секретное сообщение в некоторый контейнер, передача которого не вызывает подозрений, например, в цифровую фотографию. Если никто кроме отправителя не имеет доступ одновременно к пустому и заполненному контейнеру, то выявление наличия скрытой в нем информации может быть непростой задачей. Разработка методов выявления скрытой информации составляет предмет стегоанализа.

Стегосистемы, в которых пустые и заполненные контейнеры подчиняются одному и тому же закону распределения вероятностей (т. е. статистически неразличимы), называют совершенными [1]. В работе [2] впервые предложена простая конструкция совершенной стеганографической системы для сообщений, порожденных вероятностными источниками информации. Методы построения совершенных стегосистем рассматривались также в [3 – 5]. Основная идея, использованная при построении таких стегосистем, применяется и в нашем случае, но не для вероятностных источников, а для изображений. Используемый в настоящей работе тип контейнера может иметь внешнее представление в виде файлов форматов BMP, TIFF, PNG, PCX, TGA и др. Эти форматы строятся с применением неискажающих методов сжатия, полностью сохраняющих все биты изображения, и поэтому между ними существует взаимно-однозначное соответствие (для рассматриваемого типа изображения — RGB 24 бита). В описываемых экспериментах из-за простоты программной обработки используются BMP-файлы, но все результаты автоматически переносятся на случай сохранения изображения в файлах других указанных форматов.

*Работа выполнена в рамках НИР по Государственному контракту № 02.740.11.0396.

Максимальное количество информации, которое можно записать в контейнер при использовании некоторого метода внедрения, можно назвать *эмпирической емкостью* контейнера. На самом деле эмпирическая емкость определяется не столько контейнером (его подлинная емкость обычно не известна), сколько методом. Наиболее распространенные методы, применимые к растровым изображениям (так называемые методы LSB, встраивающие данные в младшие биты цветовых составляющих), позволяют встроить не более одного бита в один байт файла изображения. Поэтому именно такую емкость установим за 100 %. В рассматриваемом методе емкость контейнера определяется в зависимости от особенностей реализации идеи перестановок. Как правило, стеганографические программы позволяют задать степень заполнения контейнера в виде некоторой доли от его емкости (назовем ее "фактический процент заполнения"). Предполагается, что при малых значениях степени заполнения изображение меньше искажается и факт наличия встроенного сообщения определить труднее.

Первое условие, которое выполняют все стеганографические программы, состоит в том, что искажения, вносимые в контейнер при внедрении сообщения, должны быть незаметны для человека. В предлагаемом методе используется параметр, определяющий степень визуальных искажений. Чем больше значение данного параметра, тем менее заметны искажения изображения, но тем меньше становится и емкость контейнера. Таким образом, в ходе экспериментов устанавливается оптимальное значение параметра, при котором искажения визуально незаметны. Для оценок стойкости метода перестановок к стегоанализу мы применяем общеизвестный RS-анализ [6] как новое наиболее мощное средство выявления скрытой информации с общедоступной программной реализацией. В представленном методе перестановок значения бит не изменяются, незначительно меняется только порядок следования байт, поэтому статистическая структура контейнера практически не нарушается, что подтверждают результаты RS-анализа. Вместе с тем данный метод обеспечивает достаточно высокую фактическую емкость. Декодирование встроенной информации происходит за счет установленного порядка следования, который можно задать изначально в качестве секретного ключа. Без измения фактических значений бит гарантируется безошибочное извлечение информации.

В результате проведенных исследований был построен алгоритм внедрения скрытых сообщений в изображения. RS-анализ на случайной выборке из 800 изображений показал, что при уровне заполнения контейнера около 30 % предложенный алгоритм позволяет скрыть наличие встроенного сообщения в 100 % файлов, т. е. процент обнаружения равен нулю. С дугой стороны, для общедоступных стеганографических программ HIDE4PGP и STEGOTOOLS при том же уровне заполнения 30 % RS-анализ обнаружил наличие встроенной информации в 84–100 % файлов. Это демонстрирует заметное преимущество разработанного метода перед аналогами. Были также исследованы стегоконтейнеры двух типов — фотографии природных объектов, характеризующиеся плавными переходами яркостей, и фотографии преимущественно искусственных объектов, что позволило выявить ряд особенностей применения предложенного алгоритма.

## 1. Основные положения

В качестве стегоконтейнера будем рассматривать изображения, представляющие собой матрицу пикселей. Обозначим матрицу исходного изображения размером $h \times w$ через $P$ (здесь $h$ — высота изображения, или число строк в матрице, $w$ — ширина, или число

столбцов), будем начинать нумерацию элементов с единицы. Для простоты описания считаем, что $h$ и $w$ четные числа.

Строго говоря, каждый элемент изображения содержит значения трех цветовых составляющих (R, G, B). При построении алгоритма исходим из стандартного при обработке изображений предположения о независимости цветовых составляющих и производим над ними одинаковые действия. Поэтому для простоты описания условимся под элементом матрицы $P$ понимать только одну обобщенную цветовую составляющую, определяющую действия над всеми тремя.

Метод перестановок основан на свойстве близости соседних пикселей, которые в типичных изображениях почти одинаковы. Это свойство особенно характерно для изображения природных объектов, в которых нет резких переходов цвета и яркости, в отличие от искусственных объектов, например, черных линий на белом фоне. Но в фотографиях искусственных объектов также имеются свои преимущества — они состоят из однотонных областей, в которых пиксели почти не отличаются друг от друга. Для экспериментов будем использовать только оригинальные BMP-файлы. В данном случае под оригинальными BMP-файлами понимаются фотографии, взятые с цифровой фотокамеры и преобразованные в нужный формат. Преобразования, разрешенные для оригинальных файлов, включают в себя конвертирование в формат BMP с помощью программ Adobe Photoshop либо ACDSeePro3. Такое преобразование необходимо, так как большинство фотокамер записывают файлы в формате JPEG и лишь немногие в форматах TIFF и RAW. Форматы TIFF и RAW являются предпочтительными, поскольку сохраняют более полную информацию об изображении. Допустимо также уменьшение размера изображения с помощью тех же программ. Для полноты эксперимента в тестовые наборы будут входить оригинальные BMP-файлы разных размеров, преобразованные из всех перечисленных выше форматов.

Основываясь на этих рассуждениях, разделим тестовый набор файлов на два типа. Первый тип A — естественный тип фотографий, к которым относятся фотографии с плавным переходом яркостей, преимущественно природных объектов. Например, фотографии в стиле пейзаж. К типу A можно также отнести некоторые фотографии искусственных объектов, если они выглядят размыто, например, фото без вспышки при недостаточном освещении. Второй тип B — искусственный тип фотографий, к которым относятся контрастные фотографии преимущественно искусственных объектов, например, фото страниц цветного журнала. В экспериментальных исследованиях будем использовать два набора тестовых файлов — набор A, содержащий фотографии только A-типа, и набор B, содержащий фотографии только B-типа. Каждый набор содержит 400 изображений. Также будем использовать такие термины, как пустой контейнер и заполненный контейнер. Под пустым контейнером подразумевается файл, не содержащий какой-либо встроенной секретной информации. Заполненный контейнер — такой файл, в который было встроено зашифрованное стойким шифром сообщение с помощью рассматриваемого метода встраивания. Так как зашифрованное сообщение является последовательностью, практически не отличимой от случайной, то в экспериментах его будут имитировать данные, полученные с помощью генераторов псевдослучайных чисел.

Основным методом стегоанализа, который будет использован для оценки алгоритма, является метод RS [6]. RS-анализ был предложен Дж. Фридрич, М. Голяном и Р. Дю [7] и считается одним из самых эффективных. Программная реализация была произведена К. Хемпсток с поддержкой авторов настоящей статьи и прошла проверку на корректность. Анализирующая программа выдает количество встроенной информации ($L$)

в процентах от эмпирической емкости контейнера, которая высчитывается как при LSB-встраивании:

$$C_{\text{LSB}} = 3wh \text{ бит.} \tag{1}$$

Формула (1) означает, что в каждый байт пикселя встраивается один бит информации. По значению $L$ можно судить о том, заполнен был контейнер или пуст. Например, в работе [8] установлено, что при $L \geq 5\,\%$ RS-анализ классифицирует контейнер как заполненный. В применении RS-анализа будем опираться на эти сведения. В цифровой стеганографии существуют два рода ошибок [9]. Предположим, имеются две гипотезы: $H_s$ — означающая, что контейнер содержит стегосообщение, и противоположная ей гипотеза $H_c$ — означающая, что контейнер "чист" (не содержит встроенной информации). Правило решения заключается в разбиении множества наблюдений на две части так, чтобы сопоставить одну из двух гипотез каждому контейнеру. В этой задаче возможны два типа ошибок: ошибка первого рода, которая заключается в установлении гипотезы $H_s$, когда стегосообщение отсутствует, и ошибка второго рода, если принято решение $H_c$ при наличии встроенной информации.

## 2. Первая реализация метода перестановок

Первую реализацию метода перестановок можно отнести к классу LSB-методов. Идея состоит в том, что один из двух различных младших бит в двух пикселях обязательно является текущим битом сообщения и, переставив эти два бита в установленном порядке, мы встроим информацию. Возьмем матрицу изображения $P$ и будем рассматривать элементы $r$-й строки попарно: $(P_{r,1},\ P_{r,2})$, $(P_{r,3}, P_{r,4}), (P_{r,5}, P_{r,6}), \ldots, (P_{r,w-1}, P_{r,w})$. Обозначим через $q_{r,i}$ младший бит соответствующего элемента $P_{r,i}$. Пусть надо встроить сообщение $M = (m_1, m_2, \ldots, m_k)$. Если выполняется условие

$$q_{r,i} \neq q_{r,i+1}, \tag{2}$$

то для каждой пары $(P_{r,i}, P_{r,i+1})$ справедливо следующее: $q_{r,i} = m_j$ или $q_{r,i+1} = m_j$, где $m_j$ — текущий встраиваемый бит.

Установив порядок следования бита $m_j$ в паре $(P_{r,i}, P_{r,i+1})$, можно встроить его в пару. Для этого достаточно поменять местами $q_{r,i}$ и $q_{r,i+1}$. Установление порядка подразумевает соглашение о том, что первый или второй бит пары является битом встроенного сообщения. Назовем его главным битом пары. Но просто переставить два младших бита в соответствующих двух байтах было бы неправильно, так как статистические связи между младшими и старшими битами могут нарушиться. Поэтому мы переставим полностью элементы $P_{r,i}$ и $P_{r,i+1}$. Далее будет показано, что такие перестановки практически не нарушают статистическую структуру контейнера. Это значит, что в каждую пару $(P_{r,i}, P_{r,i+1})$ можно встроить один бит информации. В случае $q_{r,i} = q_{r,i+1}$ (условие (2) не выполняется) пара $(P_{r,i}, P_{r,i+1})$ просто пропускается, и при извлечении информации мы также пропускаем такую пару. Аналогичные действия производятся с каждой строкой матрицы $P$. Более формально алгоритм внедрения можно записать в следующем виде:

Главный бит пары первый:

если $q_{r,i} = m_j$, то ничего не меняем,
если $q_{r,i+1} = m_j$, то $P_{r,i} \leftrightarrow P_{r,i+1}$.

Главный бит пары второй:

$$\begin{array}{l}\text{если } q_{r,i} = m_j, \text{ то } P_{r,i} \leftrightarrow P_{r,i+1},\\ \text{если } q_{r,i+1} = m_j, \text{ то ничего не меняем.}\end{array} \qquad (3)$$

**Пример 1.** Возьмем некоторый BMP-файл и рассмотрим три пары элементов $r$-й строки матрицы $P$: (241, 246), (248, 239), (205, 187). Соответствующие младшие биты образуют последовательность (1, 0), (0, 1), (1, 1). Пусть надо встроить сообщение $M =$ (1101). Установим порядок: первый бит будет главным битом пары. В третьей паре условие (2) не выполняется, а первые две пары подходят для встраивания данных, и для них можно применить правила (3). В итоге получаем (241, 246), (239, 248), (205, 187). В результате внедрены первые два бита сообщения.

Как видно, заранее предугадать, в скольких парах будет встроена информация, нельзя, можно лишь утверждать, что это полностью зависит от контейнера.

Рассмотрим возможную попытку противника получить секретные данные, встроенные по описанной схеме. Допустим, он располагает знаниями о данном методе. Чтобы получить данные из контейнера, ему необходимо рассмотреть всего два случая. В первом случае он предположит, что главный бит пары является первым. Тогда он выделит первые биты из всех неравных пар и, возможно, получит зашифрованные данные. Во втором случае противник выделит вторые биты из всех неравных пар и также, возможно, получит зашифрованные данные. Для повышения стойкости метода необходимо менять порядок следования главного бита в паре на каждом шаге в соответствии с секретной последовательностью. Реализуется это следующим образом. С помощью того же шифра, которым шифруется встраиваемое сообщение, генерируем секретную псевдослучайную битовую последовательность (точно такая же последовательность в дальнейшем может быть получена законным получателем сообщения, знающим секретный ключ). Обозначим эту последовательность через $z = z_1, z_2, z_3, \ldots$ Элементы последовательности $z$ используются для определения порядка следования бита в паре. Например, при $z_i = 0$ главным битом в паре является первый, при $z_i = 1$ — второй бит. Следовательно, противник, располагая сведениями о методе, но не зная секретной информации, не сможет выделить из контейнера ту часть, которая является зашифрованным сообщением.

Оценим эмпирическую емкость контейнера для первой реализации метода. Напомним, что эмпирическая емкость — это максимальное количество информации, которое можно записать в контейнер при использовании некоторого метода внедрения. Это значит, что в предельном случае во всех парах выполняется условие (2). Тогда эмпирическая емкость будет равна количеству пар в матрице $P$: $C_1 = C_{\text{LSB}}/2$ бит (основываясь на формуле (1)).

Проведем экспериментальное исследование данной реализации. Как указывалось ранее, мы используем два тестовых набора файлов — A-типа (естественный тип) и B-типа (искусственный тип) размером по 400 файлов каждый.

Прежде всего необходимо проверить, насколько перестановки соседних пикселей визуально искажают изображение. Ясно: чем больше разница в яркости соседних пикселей, которые переставляются, т. е. $|P_{r,i} - P_{r,i+1}|$, тем более визуально исказится такой фрагмент изображения. Поэтому, чтобы оценить искажения в предельно худшем случае, для данного эксперимента возьмем такое изображение, на котором имеются резкие переходы яркости. Пусть это будет типичное изображение набора B. Здесь не приводятся примеры изображений, так как качество печати не позволяет рассмотреть детали,

важные при исследовании метода. Изучая изображения на широкоформатном экране монитора, мы пришли к следующему результату. На первый взгляд изображения до и после встраивания не отличить. Вместе с тем при детальном рассмотрении увеличенных фрагментов изображения на границах цветовых областей с явным переходом яркости заметны искажения. Решение этой проблемы заключается в том, что необходимо задать условие, ограничивающее перестановки, влекущие за собой искажения:

$$|P_{r,i} - P_{r,i+1}| \leq d, \tag{4}$$

где $d$ — порог искажений, значение которого выбирается экспериментально. На том же тестовом изображении будем уменьшать порог $d$ до тех пор, пока не исчезнут визуальные искажения. Здесь играет роль одно из низкоуровневых свойств человеческого зрения [9]. В табл. 1 показаны результаты визуального исследования тестового изображения после встраивания информации при различных значениях $d$. Размеры данного тестового изображения $w = 1024$, $h = 768$. Максимальная емкость, относительно которой будем оценивать процент заполнения контейнера, $C_{\text{LSB}} = 3wh = 2\,359\,296$ бит $=$ 288 Кбайт. Эмпирическая емкость метода $C_1 = C_{\text{LSB}}/2 = 144$ Кбайт. Как видно из таблицы, чем меньше значение параметра $d$, тем меньше информации помещается в контейнер и тем в меньшей мере искажено изображение. По результатам эксперимента можно установить $d = 3$. В табл. 1 приведены также данные аналогичных исследований для типичного изображения A-типа (естественный тип).

Наглядно зависимость средней фактической емкости контейнера от значения параметра $d$ приведена на рис. 1. Графики построены для изображений, которые анализировались в табл. 1. При оценке емкости на графике за 100 % принималась эмпирическая емкость данного метода $C_1$. Из приведенных кривых видно, что при увеличении параметра $d$ заполнение контейнера стремится к 50 % (относительно емкости $C_1$). Значит

Таблица 1. Исследование тестового изображения B- и A-типа после встраивания информации согласно первой реализации метода перестановок

| $d$ | Искажение изображения после встраивания | Заполнение контейнера, % |
|---|---|---|
| | *Изображения B-типа* | |
| 2–3 | Не наблюдается | 13 |
| 4–5 | При внимательном изучении некоторых фрагментов заметны трудноуловимые искажения | 19 |
| 6–8 | При внимательном изучении некоторых фрагментов заметны искажения | 22 |
| 9–255 | При рассмотрении некоторых фрагментов заметны искажения | 23–25 |
| | *Изображения A-типа* | |
| 2–5 | Не наблюдается | 11 |
| 6–8 | При внимательном изучении некоторых фрагментов заметны трудноуловимые искажения | 19 |
| 9–10 | При внимательном изучении некоторых фрагментов заметны искажения | 22 |
| 11–255 | При рассмотрении некоторых фрагментов заметны искажения | 23–25 |

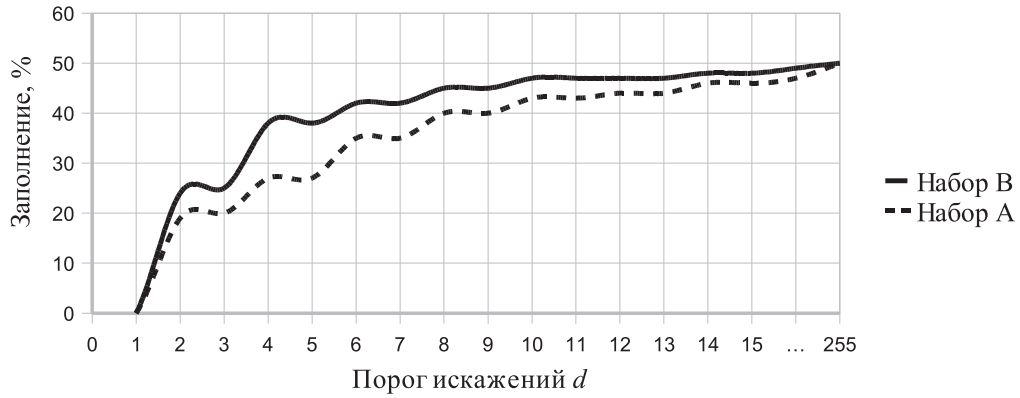


Рис. 1. Зависимость средней емкости контейнера от значения параметра $d$ относительно емкости $C_1$ для первой реализации метода перестановок

половина всех пар в данном контейнере не удовлетворяют условию (2). В дальнейшем будут предложены улучшения данного алгоритма.

Итак, установим значение параметра $d = 3$, так как при таком пороге искажения становятся незаметными на глаз для изображений как A-, так и B-типа и, с другой стороны, достигаемый уровень заполнения составляет 10–12 %, что является достаточным, поскольку RS-анализ для аналогов определяет наличие информации в контейнере уже при 5%-м заполнении. Проведем эксперимент по определению среднего процента заполнения контейнеров для изображений A- и B-типа при данном методе встраивания. В качестве зашифрованного сообщения примем случайные данные. Для полноты эксперимента в тестовые наборы A и B были включены изображения разных размеров с различной эмпирической емкостью. Процент заполнения каждого контейнера определяется в соответствии с общей эмпирической емкостью $C_{\mathrm{LSB}}$, средний процент заполнения при пороге искажений $d = 3$ для набора A составляет 10 %, для набора B — 12 %.

Полученные результаты позволяют сделать вывод о том, что изображения типа A более безопасны для встраивания информации по описанному методу, так как допустимый порог искажений для контейнеров такого типа немного больше, чем для типа B. С другой стороны, в изображения типа B помещается в среднем чуть больше информации.

Для извлечения сообщения рассматриваем матрицу $P$ таким же образом, как при встраивании информации: проверяем очередную пару на выполнение условий (2) и (4) и извлекаем младший бит из первого либо из второго байта пары, что определяется с помощью ключа, о котором говорилось выше.

Оценим стойкость метода по отношению к стегоанализу. Как уже упоминалось, в нашем случае используется метод RS. Сначала оценим процент возникновения ошибок первого рода для тестовых наборов (A и B) пустых контейнеров (табл. 2). В таблице указан процент файлов, для которых анализирующая программа обнаружила встроенную информацию в количестве $L$ %. Иначе говоря, 21 % файлов для набора A и 5 %

Т а б л и ц а 2. RS-анализ на наборе пустых контейнеров

| $L$ | 0 % | 1 – 4 % | 5 % и более |
|---|---|---|---|
| Набор A | 15 % | 64 % | 21 % |
| Набор B | 16 % | 79 % | 5 % |

Таблица 3. RS-анализ на наборе контейнеров, заполненных на 10 и 12 % соответственно для наборов A и B, с помощью известных программ

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| HIDE4PGP, набор A | 0 % | 0 % | 100 % |
| HIDE4PGP, набор B | 0 % | 0 % | 100 % |
| STEGOTOOLS, набор A | 0 % | 16 % | 84 % |
| STEGOTOOLS, набор B | 0 % | 0 % | 100 % |

Таблица 4. RS-анализ на наборе контейнеров, заполненных согласно первой реализации метода

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| Набор A | 13 % | 67 % | 20 % |
| Набор B | 23 % | 73 % | 4 % |

для набора B дают ошибку первого рода. Следовательно, фотографии природного типа значительно повышают процент ошибки. Заполним контейнеры наборов A и B случайными данными (имитация зашифрованного сообщения) с помощью общедоступных стеганографических программ HIDE4PGP и STEGOTOOLS. Программа STEGOTOOLS — последовательное LSB-заполнение, программа HIDE4PGP производит так называемое рассеянное заполнение, когда младшие биты замещаются новой информацией в порядке, определяемом некоторой псевдослучайной последовательностью. С целью сравнения результатов RS-анализа для разработанного нами метода и известных программ при равных процентах заполнения процент заполнения контейнеров в данном эксперименте был установлен соответственно указанному ранее среднему проценту заполнения. В табл. 3 приведены результаты стегоанализа для контейнеров, заполненных с помощью программ HIDE4PGP и STEGOTOOLS. Смысл данных такой же, как в предыдущей таблице. Видно, что RS-анализ обнаруживает наличие встроенной информации в 84–100 % случаев. Теперь заполним контейнеры наборов A и B случайными данными (имитация зашифрованного сообщения) согласно первой реализации метода перестановок и подвергнем их RS-анализу (табл. 4).

Результаты стегоанализа, приведенные в табл. 2 и 4, свидетельствуют о том, что процент обнаружения встроенной информации по предложенному методу практически идентичен проценту файлов, в которых имела место ошибка первого рода. Следовательно, можно утверждать, что метод устойчив к RS-атаке и может успешно использоваться. Заметим, что средняя фактическая емкость контейнера все-таки мала, что наводит на идеи дальнейшего улучшения алгоритма.

## 3. Вторая реализация метода перестановок

В предыдущей реализации метода перестановок имеется один недостаток — фактическое среднее количество помещенной информации в контейнер намного меньше эмпирической емкости контейнера. Причина этого заключается в ограничениях (2) и (4). Ограничение (4) так или иначе будет присутствовать во всех реализациях метода перестановок, так как оно обеспечивает отсутствие видимых искажений изображения при встраивании информации. Ограничение (2) рассмотрим более подробно.

Итак, согласно условию (2) перестановки не осуществляются, если младшие биты равны между собой. Но случай, когда младшие биты двух идущих подряд байт оказываются одинаковыми, встречается очень часто, однако случай равенства двух соседних байт встречается редко. Поэтому имеет смысл пересмотреть метод на возможность его модификации.

Во второй реализации метод перестановок оперирует только с байтами. Информация встраивается за счет того, что каждая пара байт упорядочена в прямом или обратном порядке, что может сопоставляться с единичным или нулевым битом соответственно. Таким образом, при рассмотрении пары байт возможны три варианта:

1) $P_{r,i} = P_{r,i+1}$ — равенство элементов;
2) $P_{r,i} < P_{r,i+1}$ — элементы расположены в прямом порядке;
3) $P_{r,i} > P_{r,i+1}$ — элементы расположены в обратном порядке.

Очевидно, что в первом случае пара не может нести информацию, так как, поменяв местами элементы, получим то же самое. Следовательно, в данной реализации метода необходимо ввести условие (5) на перестановки:

$$P_{r,i} \neq P_{r,i+1}. \tag{5}$$

Таким образом, встраивание информации в пару, для которой соблюдены условия (2) и (5), осуществляется по нижеследующим правилам (6) и (7).

Задан прямой порядок:

при $m_j = 1$:

если $P_{r,i} < P_{r,i+1}$, то ничего не меняем,

если $P_{r,i} > P_{r,i+1}$, то $P_{r,i} \leftrightarrow P_{r,i+1}$,

при $m_j = 0$:

если $P_{r,i} < P_{r,i+1}$, то $P_{r,i} \leftrightarrow P_{r,i+1}$,

если $P_{r,i} > P_{r,i+1}$, то ничего не меняем. (6)

Задан обратный порядок:

при $m_j = 0$:

если $P_{r,i} < P_{r,i+1}$, то ничего не меняем,

если $P_{r,i} > P_{r,i+1}$, то $P_{r,i} \leftrightarrow P_{r,i+1}$,

при $m_j = 1$:

если $P_{r,i} < P_{r,i+1}$, то $P_{r,i} \leftrightarrow P_{r,i+1}$,

если $P_{r,i} > P_{r,i+1}$, то ничего не меняем. (7)

**Пример 2.** Возьмем некоторый BMP-файл и рассмотрим четыре пары элементов $r$-й строки матрицы $P$: (241, 244), (248, 248), (205, 187), (189, 190). Пусть надо встроить сообщение $M = (1011)$. Установим прямой порядок следования байт в паре, т. е. будем пользоваться правилом (6). Как было установлено ранее, порог искажений $d = 3$. Проверим данные пары элементов на выполнение условий (4) и (5). Во второй паре не выполняется условие (5), в третьей паре — условие (4), а первая и третья пары подходят для встраивания данных и для них можно применить правило (7). В итоге получаем $(241, 244)$, $(248, 248)$, $(205, 187)$, $(190, 189)$. Встроено два бита сообщения.

Как и при первой реализации метода перестановок, заранее предугадать, в скольких парах будет встроена информация, нельзя, это также зависит от контейнера. Преимущество второй реализации в том, что вероятность встречаемости пары из двух одинаковых байт меньше, чем вероятность встречаемости пары байт, младшие биты которой

совпадают. За счет этого средняя фактическая емкость контейнера повышается несмотря на то, что эмпирическая емкость при первой и второй реализации одинакова, т. е. $C_2 = C_1 = C_{\mathrm{LSB}}/2$ бит.

Для повышения стойкости метода на каждом шаге в соответствии с секретной последовательностью необходимо менять порядок пары с прямого на обратный и наоборот по той же схеме, что и при первой реализации.

Проведем экспериментальное исследование метода. Вновь используем два тестовых набора файлов — A-типа (естественный тип) и B-типа (искусственный тип) размером по 400 файлов каждый.

Как и ранее, мы должны проверить, насколько данная реализация визуально искажает изображение. Порог искажений $d$ по сути должен быть одинаковым для всех реализаций метода перестановок. Однако чем больше информации встраивается в изображение, тем больше искажений получится в итоге. Рассматриваемая реализация позволяет встроить больше информации, чем первая, поэтому необходимо проверить на практике, действительно ли значение $d = 3$ применимо в данном случае метода перестановок. Для эксперимента возьмем то же изображение, которое было использовано при анализе порога для первого метода. Как было указано, максимальная и эмпирическая емкости остаются прежними — 288 и 144 Кбайт соответственно. Результаты приведены в табл. 5. Из таблицы видно, что пороговое значение $d$ можно установить таким же, как и для первой реализации.

Наглядно зависимость средней фактической емкости контейнера от параметра $d$ приведена на рис. 2. Графики построены для изображений, которые анализировались в табл. 5. При оценке емкости за 100 % принималась эмпирическая емкость данного метода $C_2$. Из приведенных кривых видно, что при увеличении параметра $d$ заполнение контейнера стремится к 100 %, но этой величины не достигает, так как в среднестатистическом изображении обязательно найдутся пары, в которых не выполняется условие (5).

Таблица 5. Исследование тестового изображения B- и A-типа после встраивания информации вторым методом

| $d$ | Искажение изображения после встраивания | Заполнение контейнера, % |
|---|---|---|
| | *Изображения B-типа* | |
| 2–3 | Не наблюдается | 18 |
| 4 | При внимательном изучении некоторых фрагментов заметны трудноуловимые искажения | 28 |
| 5–6 | При внимательном изучении некоторых фрагментов заметны искажения | 33 |
| 7–255 | При рассмотрении некоторых фрагментов заметны искажения | 36–43 |
| | *Изображения A-типа* | |
| 2–5 | Не наблюдается | 14 |
| 6 | При внимательном изучении некоторых фрагментов заметны трудноуловимые искажения | 29 |
| 7–8 | При внимательном изучении некоторых фрагментов заметны искажения | 34 |
| 9–255 | При рассмотрении некоторых фрагментов заметны искажения | 37–46 |

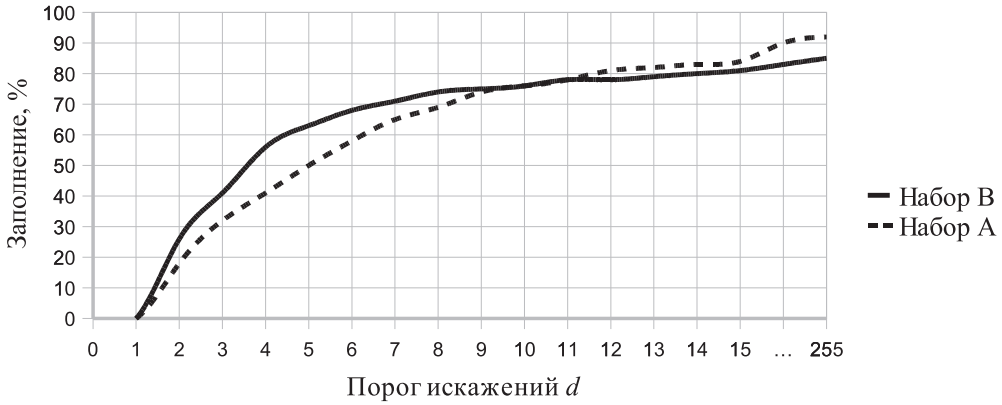


Рис. 2. Зависимость средней емкости контейнера от значения параметра $d$ относительно емкости $C_2$ для второй реализации метода перестановок

Итак, значение параметра $d$ установим опять равным 3, поскольку при таком пороге искажения становятся незаметными на глаз для изображений как A-, так и B-типа. Это свидетельствует о том, что при первой и второй реализациях метода перестановок порог искажения остается неизменным.

Проведем эксперимент по определению среднего процента заполнения контейнеров для изображений A- и B-типа. В качестве зашифрованного сообщения примем случайные данные. Для полноты эксперимента в тестовые наборы A и B были включены изображения разных размеров с различной эмпирической емкостью. Процент заполнения каждого контейнера определяется в соответствии с общей эмпирической емкостью $C_{\mathrm{LSB}}$, средние значения при установленном пороге искажений $d = 3$ для наборов A и B составляют соответственно 16 и 19 % .

Из полученных результатов следует, что изображения типа A более безопасны для встраивания информации по описанной реализации, так как допустимый порог искажений для контейнеров такого типа немного больше, чем для изображений типа B. С другой стороны, в изображения типа B помещается в среднем чуть больше информации.

Оценим устойчивость второй реализации метода к стегоанализу. Процент появления ошибок первого рода для тестовых наборов (A и B) был получен ранее и приведен в табл. 2, где показано, что 21 % файлов для набора A и 5 % для набора B дают ошибку первого рода. Заполним контейнеры наборов A и B случайными данными (имитация зашифрованного сообщения) с помощью общедоступных стеганографических программ HIDE4PGP и STEGOTOOLS. Чтобы сравнить результаты RS-анализа для разработанного нами метода и указанных программ при равных заполнениях, процент заполнения контейнеров в эксперименте установим соответственно указанному ранее среднему проценту заполнения. В табл. 6 приведены результаты стегоанализа для контейнеров, заполненных с помощью программ HIDE4PGP и STEGOTOOLS. Смысл данных такой же, как в табл. 2. Из приведенных данных видно, что RS-анализ обнаруживает наличие встроенной информации в 100 % случаев.

Заполним контейнеры наборов A и B случайными данными (имитация зашифрованного сообщения) по второй реализации рассматриваемого метода и также проведем RS-анализ (табл. 7, смысл данных тот же, что в предыдущих таблицах).

Результаты стегоанализа, приведенные в табл. 2 и 7, показывают, что процент обнаружения встроенной информации во второй реализации метода практически идентичен

Таблица 6. RS-анализ на наборе контейнеров, заполненных на 16 и 19 % соответственно для наборов A и B с помощью известных программ

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| HIDE4PGP, набор A | 0 % | 0 % | 100 % |
| HIDE4PGP, набор B | 0 % | 0 % | 100 % |
| STEGOTOOLS, набор A | 0 % | 6 % | 94 % |
| STEGOTOOLS, набор B | 0 % | 0 % | 100 % |

Таблица 7. RS-анализ на наборе контейнеров, заполненных согласно второй реализации

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| Набор A | 14 % | 65 % | 21 % |
| Набор B | 21 % | 75 % | 4 % |

проценту файлов, в которых имела место ошибка первого рода. Это значит, что в данном случае RS-анализ не обнаружил скрытой информации в контейнерах. Заметим, что во второй реализации метода перестановок удалось увеличить емкость контейнера. Дальнейшие улучшения алгоритма возможны при рассмотрении перестановок не в парах пикселей, а в отдельных группах с помощью известных методов комбинаторики.

## 4. Третья реализация метода перестановок

До сих пор рассматривалась матрица $P$ по парам элементов. Эмпирическая емкость контейнера $C_2 = C_{\text{LSB}}/2$, т. е. $1/2$ бита на один байт. На практике результат был меньше установленного максимума и полностью зависел от особенностей контейнера. В следующей, третьей, реализации метода перестановок будем рассматривать матрицу $P$ группами из $n$ элементов и обращаться к комбинаторике. Перестановками называют комбинации из $n$ различных элементов, отличающиеся только порядком расположения элементов. Известно, что число всех возможных перестановок

$$P_n = n! \tag{8}$$

Идея заключается в том, что каждой из $n!$ комбинаций можно сопоставить кодовый символ. Другими словами, мы имеем дело с задачей нумерации перестановок.

Чтобы обеспечить безошибочное декодирование, перестановки необходимо генерировать в лексикографическом порядке, т. е. самая первая комбинация всегда будет являться упорядоченной последовательностью. Заметим, однако, что в группе из $n$ элементов вполне возможны повторения. Число всех возможных перестановок с повторениями равно

$$P(n_1, n_2, \ldots, n_k) = \frac{n!}{n_1! n_2! \ldots n_k!}, \tag{9}$$

где $n_1 + n_2 + \ldots + n_k = n$, $n_k$ — количество вхождений $k$-го элемента.

Итак, первым условием в данном случае будет следующее:

$$P(n_1, n_2, \ldots, n_k) > 1. \tag{10}$$

Это значит, что группа, в которой не выполняется условие (10), остается без изменений в файле. При декодировании такие пары также пропускаются.

Второе условие для каждой группы пикселей будет накладывать ограничения на искажения при встраивании информации:

$$|P_i - P_j| \leq d \quad \text{для всех} \quad i, j = 1, \ldots, n \quad (i \neq j). \tag{11}$$

Это условие аналогично условию (4) и обеспечивает отсутствие визуальных искажений при использовании метода перестановок при заданном пороговом значении $d$. Как было установлено ранее, $d = 3$.

Далее, если группа удовлетворяет условиям (10) и (11), то для нее формируется алфавит кодовых символов $A = \{1, \ldots, m\}$, где $m = P(n_1, n_2, \ldots, n_k)$, т. е. каждый кодовый символ сопоставлен с определенной комбинацией группы. Комбинации генерируются строго в лексикографическом порядке и кодовый символ 1 всегда соответствует группе, упорядоченной в прямом порядке. Затем сформированный алфавит поступает на вход арифметического декодера. В описываемой реализации метода все символы алфавита являются равновероятными для каждой группы. Арифметическому декодеру указывается распределение вероятностей для выходных символов, в соответствии с которым он воспроизводит (декодирует) этот символ, рассматривая зашифрованное сообщение как код, построенный ранее соответствующим арифметическим кодером. Полученный символ на выходе арифметического декодера определяет номер перестановки для текущей группы. Чтобы восстановить зашифрованное сообщение из его кода, требуется, наоборот, применить арифметический кодер, которому указываются те же распределения вероятностей, что и декодеру. Таким образом, арифметическому кодеру подается на вход соответственный номер перестановки очередной группы, удовлетворяющей условиям (10) и (11), в результате чего на выходе получается декодированное сообщение. Наглядная схема третьей реализации метода приведена на рис. 3.

Поясним на примере, каким образом происходит нумерация перестановок. Возьмем произвольную группу из трех байт: 34, 32, 35. В ней не встречаются повторяющиеся элементы, поэтому количество всех возможных перестановок будет $3! = 6$. Следовательно, условие (10) выполняется. Эта группа также удовлетворяет условию (11) при установленном параметре $d = 3$. Таким образом, мы имеем алфавит $A = \{1, 2, 3, 4, 5, 6\}$ с вероятностями (1/6, 1/6, 1/6, 1/6, 1/6, 1/6) и передаем эти данные арифметическому декодеру. Так как $n$ мало, можно использовать табличные методы. Допустим, в результате работы декодера был получен символ 5. Далее мы должны заменить имеющуюся комбинацию (группу) на пятую по порядку комбинацию. Во-первых, необходимо упорядочить имеющуюся группу, установив таким образом первую комбинацию. Затем выводим в лексикографическом порядке комбинации до пятой: (32, 34, 35), (32, 35, 34), (34, 32, 35), (34, 35, 32), (35, 32, 34). Итак, текущую рабочую группу байт (34, 32, 35) заменяем пятой комбинацией (35, 32, 34).

Рассмотрим пример с группой, имеющей повторяющиеся элементы. Пусть дана группа из трех байт: 33, 35, 33. В ней два различных элемента ($k = 2$), $n_1 = 2$, $n_2 = 1$. В соответствии с (10) получаем $P_{(2,1)} = \dfrac{3!}{2!1!} = 3$, следовательно, условие (10) выполняется. Данная группа также удовлетворяет условию (11) при установленном параметре $d = 3$. Таким образом, имеем алфавит $A = \{1, 2, 3\}$ с вероятностями (1/3, 1/3, 1/3) и передаем эти данные арифметическому декодеру. Допустим, в результате работы декодера получен символ 2. Далее мы должны заменить имеющуюся комбинацию (группу) на вторую по порядку комбинацию. Прежде всего необходимо упорядочить имеющуюся группу, установив таким образом первую комбинацию. Затем выводим в лексикографи-

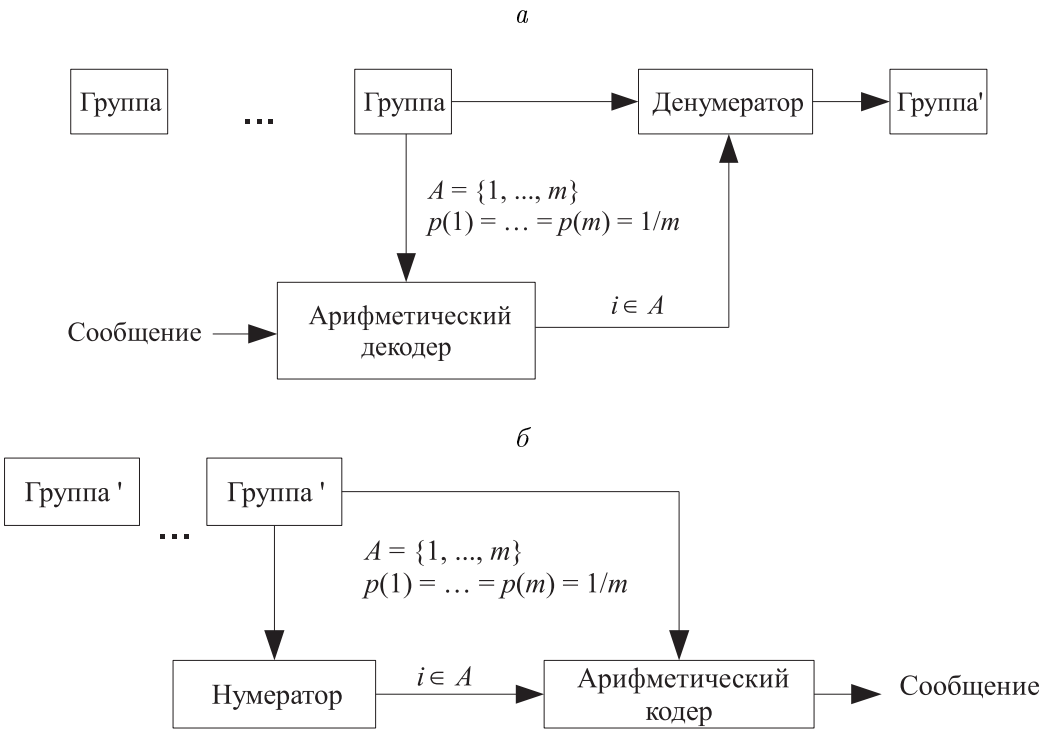


Рис. 3. Преобразование группы пикселей при внедрении (*а*) и извлечении (*б*) сообщения путем третьей реализации метода перестановок

ческом порядке комбинации до второй (33, 33, 35), (33, 35, 33). Итак, текущую рабочую группу байт (33, 35, 33) заменяем второй комбинацией (33, 35, 33). Таким образом, реализовался лучший случай, когда нужная нам комбинация совпала с изначальной группой, в результате чего группа осталась прежней, но информация была внедрена.

Проанализируем процесс извлечения информации из контейнера. Согласно схеме, представленной на рис. 3, *б*, рассматриваем матрицу по группам в том же порядке, что и при встраивании информации. Допустим, поступила текущая группа из трех байт: (12, 11, 15). В данной группе не встречаются повторяющиеся элементы, поэтому количество всех возможных перестановок будет $3! = 6$. Следовательно, условие (10) выполняется. Данная группа также удовлетворяет условию (11) при установленном параметре $d = 3$. Таким образом, выяснено, что в данную группу была встроена информация. Теперь необходимо передать арифметическому кодеру следующую информацию: алфавит, распределение вероятностей символов алфавита и номер перестановки. Мы имеем алфавит $A = \{1, 2, 3, 4, 5, 6\}$ с вероятностями (1/6, 1/6, 1/6, 1/6, 1/6, 1/6). Определим номер перестановки, соответствующий данной группе. Во-первых, упорядочим имеющуюся группу, установив таким образом первую комбинацию. Затем выводим комбинации в лексикографическом порядке до тех пор, пока одна из них не совпадет с текущей группой: (11, 12, 15), (11, 15, 12), (12, 11, 15). Итак, третья по счету комбинация совпала с текущей группой байт. Подаем на вход арифметическому кодеру полученную информацию и тем самым извлекаем секретное сообщение.

Для повышения стойкости метода необходимо менять лексикографический порядок комбинаций с прямого на обратный и наоборот на каждом шаге в соответствии с секретной последовательностью. Выполняется это по той же схеме, что и в предыдущих реализациях метода перестановок.

Остановимся подробней на том, каким образом происходит разбиение матрицы $P$ на группы. Количество групп в матрице вычисляется по формуле

$$K_g = (3wh)/n, \tag{12}$$

где $w$ — количество столбцов в матрице, $h$ — количество строк в матрице, $n$ — размер группы. В каждую группу можно встроить максимум $\log n!$ бит. Ясно: чем больше размер группы, тем выше емкость контейнера. Эмпирическая емкость метода определяется на основе формул (1) и (12) следующим образом:

$$C_n = K_g \cdot \log n! = C_{\mathrm{LSB}}/n \cdot \log n!. \tag{13}$$

При больших значениях $n$ можно применить формулу

$$\log n! \approx n \cdot \log n. \tag{14}$$

Из (13) и (14) для больших $n$ следует

$$C_n \approx C_{\mathrm{LSB}} \cdot \log n. \tag{15}$$

Заметим — это максимальное значение количества информации, которое можно встроить при допущениях о том, что каждая группа удовлетворяет условиям (10) и (11) и не имеет повторяющихся элементов. На практике такой случай возможен только при создании контейнера искусственно. В обычных контейнерах с возрастанием размера группы увеличивается и вероятность того, что такая группа не будет удовлетворять условиям (10) и (11) и будет содержать повторяющиеся элементы, что существенно уменьшит фактическую емкость. Поэтому необходимо экспериментально определить оптимальный размер группы. Для данного эксперимента сформируем группы из пикселей матрицы $P$. Вновь используем два тестовых набора файлов — A-типа (естественный тип) и B-типа (искусственный тип) размером по 400 файлов каждый и установленное пороговое значение $d = 3$. Проведем эксперимент по определению среднего процента заполнения контейнеров для изображений A- и B-типа при разных значениях $n$. В качестве зашифрованного сообщения примем случайные данные. Процент заполнения каждого контейнера определяется в соответствии с общей эмпирической емкостью $C_{\mathrm{LSB}}$, средние значения приводятся в табл. 8 при установленном пороге искажений $d = 3$. Рассмотрим подробнее схемы формирования групп. Линейная схема самая простая: элементы берутся по порядку их следования в строке. Например, при $n = 3$ группы будут следующими: $(P_{1,1}, P_{1,2}, P_{1,3})$, $(P_{1,4}, P_{1,5}, P_{1,6})$ и т. д. Группы в форме квадратов формируются соответственно указанному размеру. Например, квадраты $2 \times 2$: $(P_{1,1}, P_{1,2}, P_{2,1}, P_{2,2})$, $(P_{1,3}, P_{1,4}, P_{2,3}, P_{2,4})$ и т. д. Группы в форме треугольников при $n = 3$ формируются следующим образом: $(P_{2,1}, P_{1,1}, P_{1,2})$, $(P_{2,2}, P_{2,3}, P_{1,3})$ и т. д.

Т а б л и ц а 8. Средний процент заполнения контейнеров (третья реализация метода)

| $n$ | Схема формирования группы | Тестовый набор A, % | Тестовый набор B, % |
|---|---|---|---|
| 3 | Линейно, из идущих подряд пикселей | 28 | 34 |
| 4 | Линейно, из идущих подряд пикселей | 27 | 34 |
| 3 | В форме треугольников | 18 | 20 |
| 4 | В форме квадратов $2 \times 2$ | 23 | 32 |

Таблица 9. RS-анализ на наборах контейнеров A и B, заполненных на 28 и 34 % соответственно, с помощью известных программ

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| HIDE4PGP, набор A | 0 % | 0 % | 100 % |
| HIDE4PGP, набор B | 0 % | 0 % | 100 % |
| STEGOTOOLS, набор A | 0 % | 3 % | 97 % |
| STEGOTOOLS, набор B | 0 % | 0 % | 100 % |

Таблица 10. RS-анализ на наборах контейнеров A и B, заполненных согласно третьей реализации

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| Набор A | 15 % | 65 % | 20 % |
| Набор B | 15 % | 75 % | 10 % |

Таблица 11. Сводная таблица результатов RS-анализа (три реализации метода перестановок)

| $L$ | 0 % | 1–4 % | 5 % и более |
|---|---|---|---|
| Набор A. Ошибка первого рода | 15 % | 64 % | 21 % |
| Набор A. Первая реализация (10 %) | 13 % | 67 % | 20 % |
| Набор A. Вторая реализация (16 %) | 14 % | 65 % | 21 % |
| Набор A. Третья реализация (28 %) | 15 % | 65 % | 20 % |
| Набор B. Ошибка первого рода | 16 % | 79 % | 5 % |
| Набор B. Первая реализация (12 %) | 23 % | 73 % | 4 % |
| Набор B. Вторая реализация (19 %) | 21 % | 75 % | 4 % |
| Набор B. Третья реализация (34 %) | 21 % | 75 % | 4 % |

Итак, с помощью нумерации перестановок мы увеличили среднюю емкость контейнеров, сохраняя допустимый порог искажений. Из данных табл. 8 следует, что увеличение размера группы не приводит к лучшим результатам и при линейном порядке формирования группы целесообразно установить $n = 3$.

Оценим стойкость третьей реализации метода к стегоанализу. Процент возникновения ошибок первого рода для тестовых наборов (A и B) был приведен в табл. 2, где показано, что 21 % файлов для набора A и 5 % для набора B дают ошибку первого рода. Заполним контейнеры наборов A и B случайными данными с помощью общедоступных стеганографических программ HIDE4PGP и STEGOTOOLS. Процент заполнения контейнеров, как и в оценке предыдущих реализаций, был установлен соответственно указанному среднему проценту заполнения с целью сравнения результатов RS-анализа для разработанного метода и указанных программ при равных процентах заполнения. В табл. 9 приведены результаты стегоанализа для контейнеров, заполненных с помощью программ HIDE4PGP и STEGOTOOLS. Смысл данных такой же, как в табл. 2. Из табл. 9 видно, что RS-анализ обнаруживает наличие встроенной информации в 100 % случаев.

Заполним контейнеры наборов A и B случайными данными согласно третьей реализации и оценим результаты RS-анализа (табл. 10). Из данных табл. 2 и 10 следует, что процент обнаружения встроенной информации согласно третьей реализации метода

перестановок приблизительно равен проценту файлов, в которых имела место ошибка первого рода, что свидетельствует о стойкости данной реализации к RS-атаке. Таким образом, при уровне внедрения информации до 34 % от емкости заполненного контейнера последний не удается надежно отличить от пустого. Сравнение результатов стегоанализа трех реализаций метода перестановок, приведенное в табл. 11, показывают преимущества разработанного метода над известными аналогами.

## Список литературы

[1] Cachin C. An information-theoretic model of steganography // Lecture Notes in Computer Science (Proc. 2nd Information Hiding Workshop). Springer-Verlag, 1998. Vol. 1525. P. 306–318.

[2] Ryabko B., Ryabko D. Information-theoretic approach to steganographic systems // IEEE Intern. Symp. on Information Theory. Nice, France, 2007. P. 2461–2464.

[3] Fionov A., Ryabko B. Simple ideal steganographic systems for containers with known statistics // XI Intern. Symp. on Problems of Redundancy. St.-Petersburg, 2007. P. 184–188.

[4] Рябко Б.Я., Фионов А.Н. Алгоритмы кодирования для идеальных стеганографических систем // Вестник НГУ. Информ. технологии. 2008. № 2. С. 88–93.

[5] Hoshi M., Han T.S. Interval algorithm for homophonic coding // IEEE Trans. Inform. Theory. 2001. Vol. 47. P. 1021–1031.

[6] Рябко Б.Я., Рябко Д.Б. Асимптотически оптимальные совершенные стеганографические системы // Проблемы передачи информации. 2009. Т. 45, вып. 2. С. 119–126.

[7] Fridrich J., Goljan M., Du R. Reliable detection of LSB steganography in color and grayscale images // Proc. of the ACM Workshop on Multimedia and Security. Ottawa, Canada, 2001. P. 27–30.

[8] Жилкин М.Ю. Теоретико-информационные методы стегоанализа графических данных. Дисс. ... канд. техн. наук. Новосибирск, СибГУТИ, 2009.

[9] Грибунин В.Г., Оков И.Н., Туринцев И.В. Цифровая стеганография. М.: Солон-Пресс, 2002. 272 с.

*Поступила в редакцию 13 ноября 2010 г.,*
*с доработки — 11 января 2011 г.*