

ИЗ ФОНДОВ РОССИЙСКОЙ ГОСУДАРСТВЕННОЙ БИБЛИОТЕКИ

Балакин, Александр Владимирович

**Разработка архитектуры программного комплекса
и методов информационной защиты
мультимедиа-информации с использованием
цифровых водяных знаков**

Москва

Российская государственная библиотека

Балакин, Александр Владимирович

Разработка архитектуры программного комплекса и методов информационной защиты мультимедиа-информации с использованием цифровых водяных знаков : [Электронный ресурс] : Дис. ... канд. техн. наук : 05.13.11, 05.13.19. - Ростов н/Д: РГБ, 2006 (Из фондов Российской Государственной Библиотеки)

Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Методы и системы защиты информации, информационная безопасность

Полный текст:

<http://diss.rsl.ru/diss/06/0151/060151009.pdf>

**Текст воспроизводится по экземпляру, находящемуся в
фонде РГБ:**

Балакин, Александр Владимирович

*Разработка архитектуры программного
комплекса и методов информационной защиты
мультимедиа-информации с использованием
цифровых водяных знаков*

Ростов н/Д 2005

61:06-5/817

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ НАУЧНОЕ УЧРЕЖДЕНИЕ НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ
"СПЕЦИАЛИЗИРОВАННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ УСТРОЙСТВА ЗАЩИТЫ И АВТОМАТИКА"

На правах рукописи



Балакин Александр Владимирович

**РАЗРАБОТКА АРХИТЕКТУРЫ ПРОГРАММНОГО КОМПЛЕКСА
И МЕТОДОВ ИНФОРМАЦИОННОЙ ЗАЩИТЫ МУЛЬТИМЕДИА-
ИНФОРМАЦИИ С ИСПОЛЬЗОВАНИЕМ
ЦИФРОВЫХ ВОДЯНЫХ ЗНАКОВ**

Специальности:

05.13.11 – Математическое и программное обеспечение вычислительных машин,
комплексов и компьютерных сетей

05.13.19 – Методы и системы защиты информации, информационная безопасность

Диссертация на соискание ученой степени
кандидата технических наук

Научные руководители:

доктор технических наук,
профессор

Аграновский А.В.

кандидат технических наук,
Хади Р.А.

СОДЕРЖАНИЕ

Принятые обозначения и сокращения	5
Введение	6
Глава 1. Исследование и анализ существующих средств защиты и методов контроля целостности мультимедиа-информации.....	14
1.1. Анализ существующих и потенциальных угроз безопасности мультимедиа-информации	15
1.2. Исследование и анализ существующих методов внедрения цифровых водяных знаков.....	19
1.2.1. Анализ существующих методов внедрения цифровых водяных знаков	20
1.2.2. Исследование методов оценки стойкости систем внедрения информации.....	24
1.3. Обзор и анализ существующих методов защиты и контроля целостности мультимедиа-информации	37
1.3.1. Анализ архитектуры программных систем защиты и контроля целостности информации.....	37
1.3.2. Анализ методов защиты информации	39
1.3.3. Анализ современных методов контроля целостности	43
1.4. Постановка задачи исследования	46
1.5. Выводы.....	49
Глава 2. Разработка математических моделей и методов контроля целостности мультимедиа-информации путем скрытого внедрения и извлечения цифровых водяных знаков.....	51
2.1. Разработка модели естественности мультимедиа-информации.....	52
2.1.1. Определение пространства сокрытия в мультимедиа-информации	52
2.1.2. Разработка базовой модели естественности мультимедиа-информации.....	55
2.1.3. Параметры контроля пар значений различных элементов пространства сокрытия	57
2.1.4. Параметры контроля частот серий битовых значений различных отсчетов контейнера	61
2.1.5. Параметры контроля длин битовых серий различных отсчетов контейнера...64	

2.2. Разработка методов внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров	67
2.3. Разработка метода контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков	74
2.4. Разработка модели оценки вносимых при внедрении искажений и методов их количественной оценки	75
2.5. Выводы	80
Глава 3. Архитектура программного комплекса контроля целостности мультимедиа-информации	82
3.1. Разработка обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации	84
3.1.1. Агент контроля и управления.....	86
3.1.2. Агент информационного хранилища.....	87
3.1.3. Агент управления.....	88
3.1.4. Агент регистрации мультимедиа-информации	89
3.1.5. Агент клонирования мультимедиа-информации.....	92
3.1.6. Агент анализа и контроля целостности.....	92
3.2. Разработка универсальных интерфейсов взаимодействия управляющих и функциональных компонент	93
3.3. Разработка открытых интерфейсов для подключения программного обеспечения стороннего разработчика.....	95
3.3.1. Интерфейс для подключения модулей для работы с мультимедиа объектами IMediaStream	96
3.3.2. Интерфейс для подключения модулей выделения пространства сокрытия IMediaContainer	100
3.3.3. Интерфейс для подключения модулей работы с массивами мультимедиа потоков IArrayOfMediaStream	103
3.4. Выводы.....	107
Глава 4. Практические аспекты реализации программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков.....	109

4.1. Реализация программного обеспечения функциональных компонент, реализующих работу с мультимедиа-контейнерами, сокрытие и извлечение цифровых водяных знаков и контроль целостности.....	110
4.1.1. Реализация функциональных модулей, использующих универсальные интерфейсы взаимодействия управляющих и функциональных компонент	111
4.1.2. Реализация функциональных модулей, использующих открытые интерфейсы для подключения программного обеспечения стороннего разработчика	118
4.1.3. Реализация интерфейсов для взаимодействия с пользователем	133
4.2. Экспериментальная оценка аспектов функционирования комплекса, вносимых искажений и нарушения естественности мультимедиа-контейнеров	136
4.3. Выводы.....	144
Заключение	145
Литература.....	150
Приложение А. Обзор запатентованных решений в области внедрения цифровых водяных знаков.....	163
Приложение Б. Исходные тексты программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков..	173
Приложение В. Акты об использовании результатов диссертационной работы	190

ПРИНЯТЫЕ ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Z_2^n – множество двоичных строк длины n

$|M|$ мощность множества M

$\{0,1\}$ множество из 0 и 1, то есть $|\{0, 1\}| = 2$

$\{0,1\}^n$ (битовая) строка длины n , состоящая из элементов $\{0, 1\}$

$\{0,1\}^*$ битовая строка произвольной длины

$|b|$ длина битовой строки b

$a||b$ конкатенация (дописывание) строки b к строке a

$\lfloor x \rfloor$ максимальное целое число, меньшее x

$\lceil x \rceil$ минимальное целое число, большее x

ЦВЗ цифровой водяной знак

ПС пространство сокрытия

ИПС используемое пространство сокрытия

КС коэффициент сокрытия

КЭ коэффициент эффективности метода сокрытия

ГСЧ – генератор случайных чисел

ДКП – дискретное косинусное преобразование

ГПСЧ – генератор псевдо-случайных чисел

АЦП – аналого-цифровое преобразование

ЦАП – цифро-аналоговое преобразование

АЧХ – амплитудно-частотная характеристика

ВВЕДЕНИЕ

Стеганография – это наука о способах передачи (хранения) скрытой информации, при которых скрытый канал организуется на базе и внутри открытого канала с использованием особенностей его организации. Основными научными направлениями современной стеганографии является изучение способов и методов сокрытия секретных сообщений, а также изучение методов анализа стойкости и раскрытия стеганографических систем.

Актуальность темы. Благодаря появлению большого числа прикладных задач, связанных с необходимостью построения анонимных каналов передачи данных (например, в системах он-лайновых и Интернет-платежей, комплексах получения доступа к информационным ресурсам) и обеспечением защиты авторских прав в сфере информационных технологий, научные достижения в области стеганографии стали представлять особый практический интерес.

Не так давно стеганография представляла интерес в основном для военных и дипломатических кругов. Сегодня, в силу целого ряда обстоятельств, резко повысился интерес к применению стеганографии в коммерческой области.

В настоящее время резко возросло количество различной информации передаваемой в цифровой форме, появились электронные средства массовой информации, осуществляется цифровое спутниковое телевещание, распространяются цифровые сети мобильной связи. Все это, с одной стороны, способствует развитию и построению принципов организации анонимных каналов передачи данных, а с другой стороны, требует развития методов, позволяющих обеспечить эффективную защиту авторских и имущественных прав на цифровую информацию, передаваемую по сетям общего пользования.

Методы стеганографии позволяют вести скрытый обмен информацией в мировых масштабах с использованием коммуникационных возможностей сети Интернет. Поэтому в последнее время особое значение приобретают методы стеганоанализа, позволяющие производить обнаружение и уничтожение скрытой информации, а также проводить общий анализ существующих стеганографических систем.

Резкое увеличение объемов хранимой и обрабатываемой информации делает задачу идентификации в хранилищах электронных данных крайне актуальной. Резко возросший объем носителей информации, позволяет создавать большие архивы мультимедийной информации (аудиозаписей, видеороликов, графических изображений). В связи с этим перспективным является применение методов стеганографии для упрощения процесса создания и работы с подобными хранилищами. Например, при поступлении новых материалов в них внедряется заголовок, содержащий их номер, название, описание, цифровые водяные знаки и т.п., что позволяет значительно упростить поиск и идентификацию, а так же легко определить принадлежность тех или иных данных.

Увеличение количества публикуемых работ и патентуемых решений, связанных с вопросами стеганографии, несомненно, указывает на увеличение темпов развития данной науки. Однако, анализ существующих на сегодняшний день стеганографических систем показывает, что они не могут полностью удовлетворить требованиям существующих практических задач, обеспечить требуемый уровень скрытности, целостности и надежности внедрения данных.

Более того, на сегодняшний день не существует стеганографической системы, которая могла бы решить одновременно задачу контроля целостности, задачу контроля за различными копиями одного и того же мультимедиа-контейнера и задачу классификации различных мультимедиа-контейнеров в хранилищах мультимедийных данных. Поэтому, задача разработки подобной системы является актуальной как с научной, так и с практической точки зрения.

Для преодоления указанных недостатков современных стеганографических систем требуется разработка моделей, методов, алгоритмов и программных инструментальных средств организации систем контроля целостности, контроля распространения различных копий одного и того же мультимедиа-контейнера и его классификации в условиях существующих угроз безопасности мультимедиа-информации, что представляет собой научную проблему.

Объект исследования: распределенные программные системы контроля целостности мультимедиа-контейнеров, основанные на внедрении и извлечении цифровых водяных знаков.

Предмет исследования: методы и алгоритмы сокрытия цифровых водяных знаков в стеганографические мультимедиа-контейнеры обеспечивающие классификацию,

идентификацию и контроль целостности на основе сокрытых данных в условиях существующих угроз безопасности мультимедиа-информации, архитектура распределенных систем контроля целостности мультимедиа-информации.

Целью диссертационной работы является разработка архитектуры и расширение функциональности программного обеспечения защиты информации для повышения информационной защищенности мультимедиа-информации при ее передаче, обработке и хранении в информационно-вычислительных сетях.

Исходя из поставленной цели, основной научной задачей является: разработка программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров.

Основная задача включает следующие этапы решения:

- анализ существующих и потенциальных угроз безопасности мультимедиа-информации, методов сокрытия данных и методов контроля их целостности;
- разработка модели и методов внедрения и извлечения цифровых водяных знаков, и контроля целостности мультимедиа-контейнеров;
- разработка модели оценки вносимых при внедрении цифровых водяных знаков искажений и методов их количественной оценки;
- разработка обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации, включающей интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения программного обеспечения стороннего разработчика.

Методы исследования основаны на использовании теорий вероятности и математической статистики, цифровой фильтрации сигналов, математической теории связи, математического моделирования, численного эксперимента.

Границы исследования. В работе рассматриваются вопросы организации систем контроля целостности мультимедиа-информации посредством применения разрабатываемых методов внедрения специальных цифровых водяных знаков в условиях активного противодействия (поиска, обнаружения, извлечения и подмены сокрытых цифровых водяных знаков).

Научная новизна заключается в новом подходе к организации сокрытия и извлечения информации, основанном на предложенном формате скрываемого сообщения, позволяющем использовать разработанную систему внедрения цифровых водяных знаков для контроля целостности, задач классификации и идентификации; в предложенных модельных решениях построения процедур сокрытия и извлечения информации, позволяющих использовать в качестве контейнеров мультимедиа-контейнеры; в предложенной архитектуре программного комплекса, позволяющей обеспечить гибкое управление функциональностью разработанной системы.

Основные положения и научные результаты, выносимые на защиту:

1. Методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров зависящие от секретного ключа, однозначно определяющего способ внедрения и извлечения информации, отличающиеся от известных тем, что они разработаны на основе модели естественности мультимедиа-информации.
2. Впервые разработанный метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков, позволяющий использовать существующие хэш-функции и проводить контроль целостности только при наличии мультимедиа-контейнера, секретного ключа и возможно ключа хэш-функции.
3. Метод количественной оценки вносимых при внедрении цифровых водяных знаков искажений, отличающийся от известных тем, что для оценки используется не одна метрика, а специально сформированный набор метрик.
4. Впервые разработанная обобщенная архитектура программного комплекса контроля целостности мультимедиа-информации основанная на мультиагентном подходе, позволяющая осуществлять масштабирование комплекса и его настройку в соответствии с уже имеющейся информационно-вычислительной инфраструктурой.

5. Открытые интерфейсы для подключения программного обеспечения стороннего разработчика, отличающиеся тем, что они позволяют изменять функциональность отдельных агентов и комплекса в целом за счет разработки и подключения модулей выделения пространства сокрытия и работы с форматами хранения мультимедиа-информации.

Практическая ценность исследования заключается в возможности использования разработанной программной архитектуры, включающей интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения

программного обеспечения стороннего разработчика при проектировании и реализации систем контроля целостности мультимедиа-информации нового типа, основанных на методах внедрения и извлечения цифровых водяных знаков. Это позволяет не только существенно упростить процесс разработки, тестирования, эксплуатации и модернизации программного обеспечения, но и избавиться от необходимости организации хранения и распределения кодов контроля целостности. Результаты диссертационного исследования могут быть использованы при проектировании и реализации программных и аппаратно-программных систем защиты мультимедиа-информации для обеспечения информационной безопасности в финансовых, банковских, ведомственных и промышленных электронно-вычислительных комплексах и сетях.

Основные результаты исследований использованы при выполнении хоздоговорных научно-исследовательских работ и внедрены в учебный процесс:

- НИР "Бородино", "Ключ"; ОКР "Заслонка" в ФГНУ НИИ "Спецвузавтоматика" (г. Ростов-на-Дону);
- НИР "Полтава-ТС" в МТУСИ (г. Москва);
- НИР "Картина-А" в ГНИИИ ПТЗИ ФСТЭК России (г. Воронеж);
- учебные курсы "Криптография и специальные исследования" и "Основы криптографии" в Саровском государственном физико-техническом институте РФЯЦ ВНИИЭФ (г. Саров).

Апробация работы. Основные результаты диссертационной работы докладывались и обсуждались на российских и международных научных конференций (в том числе на международной научно-методической конференции "Телематика'2001" г. Москва, на международной научной конференции "Интеллектуальные и многопроцессорные системы-2001" г. Таганрог, на международной научной конференции "Искусственный интеллект. Интеллектуальные многопроцессорные системы-2004" г. Таганрог, на четвертой международной конференции "Комплексная защита информации" г. Сузdalь, на научной конференции "Безопасность информационных технологий" г. Пенза, на международной конференции "Проблемы управления безопасностью сложных систем" г. Москва, на пятой международной конференции "РусКрипто-2003" г. Москва, на Всероссийской научной конференции "Научный сервис в сети Интернет" г. Новороссийск).

Авторство, новизна и полезность принципиальных технических решений защищены тремя патентами РФ и тремя свидетельствами об официальной регистрации программных продуктов.

Публикации. По теме диссертации опубликовано 40 научных трудов, из которых 4 опубликованы единолично, в том числе 2 монографии, 3 патента РФ, 3 официальных свидетельства о регистрации программ в реестре Федерального агентства РФ по патентам и товарным знакам и 9 научных статей в центральных научных журналах (в том числе в журналах, включенных в перечень ВАК).

Структура и объем диссертации. Диссертация состоит из введения, четырех глав, заключения, списка литературы и приложений. Работа содержит 149 страниц основного текста, 34 страницы приложений и включает 53 рисунка. Список литературы состоит из 129 наименований.

В первой главе диссертационной работы исследуются существующие средства защиты и методы контроля целостности мультимедиа информации. В первом параграфе главы исследуются существующие угрозы безопасности мультимедиа-информации, в нем рассмотрены первичные классы угроз и система классификационных признаков, позволяющая проводить анализ угроз в случаях, когда невозможно формально задать все элементы множества угроз. Второй параграф главы посвящен обзору и анализу существующих методов внедрения цифровых водяных знаков. В нем приведены современные термины и модели внедрения информации, описаны результаты анализа запатентованных решений в области внедрения цифровых водяных знаков и сформулированы основные недостатки данных методов. Рассмотрены методы оценки стойкости систем внедрения информации, необходимые для разработки методов внедрения и их последующего анализа. В третьем параграфе произведен анализ существующих симметричных криптографических преобразований и методов контроля целостности мультимедиа-информации. В нем проводится обзор и анализ криптографических преобразований, которые используются при построении систем защиты мультимедиа-информации, а именно симметричных шифров, бесключевых хэш-функций, хэш-функций с зависимостью от секретного ключа. На основании первых трех параграфов, в четвертом параграфе производится постановка задачи исследования. В нем формулируется основная задача исследования, на основании которой ставятся частные

задачи исследования, последовательное решение которых позволит достичь основной цели.

Во второй главе на основе проведенных научных исследований разрабатываются математические модели и методы контроля целостности мультимедиа-информации путем скрытого внедрения и извлечения цифровых водяных знаков. Решению задачи разработки модели естественности мультимедиа-информации, предназначеннной для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков, посвящен первый параграф. Отличительной особенностью разрабатываемой модели естественности является понятие пространства сокрытия в мультимедиа информации, которое позволяет использовать модель для различных видов и форматов хранения мультимедиа-информации. Затем во втором параграфе на основе разработанной модели естественности решается задача разработки методов внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров. Данные методы зависят от секретного ключа, однозначно определяющего способ внедрения и извлечения информации и позволяют равномерно распределять внедряемую информацию по пространству сокрытия. В третьем параграфе разрабатывается метод контроля целостности мультимедиа-контейнеров, основанный на разработанном методе внедрения. Данный метод в корне отличается от существующих тем, что при помощи использования стандартных хэш-функций он позволяет проводить контроль целостности при наличии только мультимедиа-контейнера и секретного ключа. Решению задачи разработки модели оценки вносимых при внедрении цифровых водяных знаков искажений и методов их количественной оценки посвящен четвертый параграф. Данная модель необходима для получения разносторонних численных характеристик, позволяющих оценить ухудшения качества мультимедиа-информации.

В третьей главе разрабатывается архитектура программного комплекса контроля целостности мультимедиа-информации. Первый параграф данной главы посвящен описанию разработанной обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации, основанной на использовании программных агентов, реализующих разработанные методы внедрения ЦВЗ и контроля целостности мультимедиа-информации. Во втором параграфе главы приведено описание системы универсальных интерфейсов взаимодействия управляющих и функциональных компонент, разработанных в соответствии с разработанной в первой части обобщенной архитектурой

программного комплекса контроля целостности мультимедиа-информации и позволяющих производить подключение общих функциональных модулей к различным типам программных агентов. В третьем параграфе проводится разработка открытых интерфейсов для подключения программного обеспечения стороннего разработчика, позволяющих производить изменения и наращивание функциональности комплекса в целом и отдельных его программных агентов за счет подключения модулей сторонних разработчиков.

Четвертая, заключительная, глава посвящена практическим аспектам реализации программного комплекса контроля целостности мультимедиа-информации. В ней приводятся особенности реализации главного модуля программного комплекса, а также программного обеспечения функциональных компонент, реализующих работу с мультимедиа-контейнерами, сокрытие и извлечение цифровых водяных знаков и контроль целостности. Проводится оценка вносимых искажений и нарушения естественности мультимедиа-контейнеров и эффективности по скорости, памяти, вычислительным ресурсам, вносимым искажениям и нарушению естественности мультимедиа-контейнеров.

В заключении обобщены итоги и результаты проведенных исследований.

Глава 1. Исследование и анализ существующих средств защиты и методов контроля целостности мультимедиа-информации

В данной главе диссертационной работы исследуются существующие средства защиты и методы контроля целостности мультимедиа информации.

В первом параграфе главы проведен анализ существующих угроз безопасности мультимедиа-информации, в нем рассмотрены первичные классы угроз и система классификационных признаков, позволяющая проводить анализ угроз в случаях, когда невозможно формально задать все элементы множества угроз.

Второй параграф главы посвящен обзору и анализу существующих методов внедрения цифровых водяных знаков. В нем приведены современные термины и модели внедрения информации, описаны результаты анализа запатентованных решений в области внедрения цифровых водяных знаков и сформулированы основные недостатки данных методов. Рассмотрены методы оценки стойкости систем внедрения информации, необходимые для разработки методов внедрения и их последующего анализа.

В третьем параграфе произведен анализ существующих симметричных криптографических преобразований и методов контроля целостности мультимедиа-информации. В нем проводится обзор и анализ криптографических преобразований, которые используются при построении систем защиты мультимедиа-информации, а именно симметричных шифров, бесключевых хэш-функций, хэш-функций с зависимостью от секретного ключа.

На основании первых трех параграфов, в четвертом параграфе производится постановка задачи исследования. В нем формулируется основная задача исследования, на основании которой ставятся частные задачи исследования, последовательное решение которых позволит достичь основной цели.

Разработка программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров включает в себя создание системы внедрения и извлечения ЦВЗ. Изучением данных систем занимается стеганография. Основными стеганографическими понятиями являются *сообщение* и *контейнер*. Сообщением m называют секретную информацию, наличие которой необходимо скрыть. Всевозможные сообщения объединяются в пространство сообщений $M \ni m$.

Контейнером $b \in B$ называют несекретную информацию, которую используют для скрытия сообщений, где B – множество всех контейнеров. Секретный ключ k представляет собой некоторую секретную информацию, известную только законному пользователю. Через K обозначается множество всех допустимых секретных ключей. В общем случае в качестве сообщений, контейнеров и ключей могут быть использованы объекты произвольной природы. Пустой контейнер (или, еще говорят, немодифицированный контейнер) – это некоторый контейнер b , не содержащий сообщения. Заполненный контейнер (или соответственно модифицированный контейнер) – это контейнер b , содержащий сообщение m , в дальнейшем b_m .

Стеганографическим преобразованием над ними принято называть зависимости $F: M \times B \times K \rightarrow B$ и $F^1: B \times K \rightarrow M$, сопоставляющие тройке (сообщение, пустой контейнер, ключ) контейнер результат и паре (заполненный контейнер, ключ) – исходное сообщение, т.е.

$$F(m, b, k) = b_{m,k}, F^1(b_{m,k}) = m, \text{ где } m \in M; b, b_{m,k} \in B; k \in K.$$

Стеганографической системой называют (F, F^1, M, B, K) совокупность сообщений, секретных ключей, контейнеров и связывающих их преобразований.

1.1. Анализ существующих и потенциальных угроз безопасности мультимедиа-информации

Под угрозой безопасности информации (компьютерной системы) принято понимать потенциально возможное воздействие на информацию (компьютерную систему), которое прямо или косвенно может нанести урон пользователям или владельцам информации (компьютерной системы) [105, 51].

В соответствии с данным определением угрозой безопасности мультимедиа-информации является потенциально возможное воздействие на мультимедиа-информацию, которое прямо или косвенно может нанести урон пользователям или владельцам мультимедиа-информации.

Основой любой классификации угроз являются три основные свойства безопасности информации [105]:

- конфиденциальность информации – субъективно определяемая характеристика информации, указывающая на необходимость введения ограничений на круг субъектов имеющих доступ к данной информации;

- целостность информации – свойство информации, заключающееся в ее существовании в неискаженном виде (неизменном по отношению к некоторому фиксированному ее состоянию);

- доступность информации – свойство компьютерной системы (среды средств и технологий обработки), в которой циркулирует информация, характеризующееся способностью обеспечивать своевременный беспрепятственный доступ субъектов к интересующей их информации и готовность соответствующих автоматизированных систем к обслуживанию поступающих от субъектов запросов всегда, когда в обращении к ним возникает необходимость.

В соответствии с этими свойствами безопасности информации выделяют три вида угроз:

- угроза конфиденциальности информации состоит в нарушении установленных ограничений на доступ к информации;

- угроза целостности информации - несанкционированное изменение информации случайное или преднамеренное.

- угроза доступности информации осуществляется, когда несанкционированно блокируется доступ к информации (блокирование может быть временным или постоянным).

К основным свойствам безопасности мультимедиа-информации можно отнести только первые два свойства:

- конфиденциальность мультимедиа-информации;
- целостность мультимедиа-информации.

Свойство доступности нельзя отнести непосредственно к самой мультимедиа-информации, оно в большей степени связано с сопутствующими системами обработки и обеспечения информационной защиты.

Таким образом, основными типами угроз мультимедиа информации являются:

- угроза конфиденциальности мультимедиа-информации;
- угроза целостности мультимедиа-информации.

Возможность реализации угроз в информационных системах зависит от наличия в них уязвимых мест. Состав и специфика уязвимых мест определяется видом решаемых задач, характером обрабатываемой информации, аппаратно-программными особенностями системы, наличием средств защиты, их характеристиками и т. д.

Однако архитектура современных средств автоматизированной обработки мультимедиа-информации, организационное, структурное и функциональное построение информационно-вычислительных систем и сетей, технологии и условия автоматизированной обработки информации приводят зачастую к тому, что накапливаемая, хранимая и обрабатываемая мультимедиа-информация подвержена случайным влияниям чрезвычайно большого числа факторов. В таких условиях становится невозможным формализовать задачу описания полного множества существующих и потенциальных угроз. Как следствие для защищаемой информации определяют не полный перечень угроз, а детализированную классификацию угроз.

Основываясь на анализе результатов научных исследований и практических разработок [74, 97, 99, 121, 124] можно выделить основные классификационные признаки угроз безопасности мультимедиа-информации:

- по степени риска;
- по природе возникновения;
- по степени преднамеренности проявления;
- по непосредственному источнику угроз;
- по положению источника угроз;
- по степени воздействия.

Использование классификационного признака "*по степени риска*" порождает следующие классы:

- угрозы с высокой степенью риска;
- угрозы со средней степенью риска;
- угрозы с низкой степенью риска.

Использование классификационного признака "*по природе возникновения*" порождает следующие классы:

- естественные угрозы – угрозы вызванные воздействиями объективных физических процессов или стихийных природных явлений, независящих от человека;
- искусственные угрозы – угрозы, вызванные деятельностью человека.

Использование классификационного признака "*по степени преднамеренности проявления*" порождает следующие классы:

- угрозы случайного действия – угрозы вызванные ошибками или халатностью персонала;

- угрозы преднамеренного действия (например, угрозы действий злоумышленника для хищения информации).

Использование классификационного признака "*по непосредственному источнику угроз*" порождает следующие классы:

- угрозы, непосредственным источником которых является природная среда;
- угрозы, непосредственным источником которых является человек;
- угрозы, непосредственным источником которых являются санкционированные программно-аппаратные средства.
- угрозы, непосредственным источником которых являются несанкционированные программно-аппаратные средства.

Использование классификационного признака "*по положению источника угроз*" порождает следующие классы:

- внешние угрозы;
- внутренние угрозы.

Использование классификационного признака "*по степени воздействия*" порождает следующие классы:

- пассивные угрозы;
- активные угрозы.

Общая схема классификационных признаков представлена на рисунке 1.

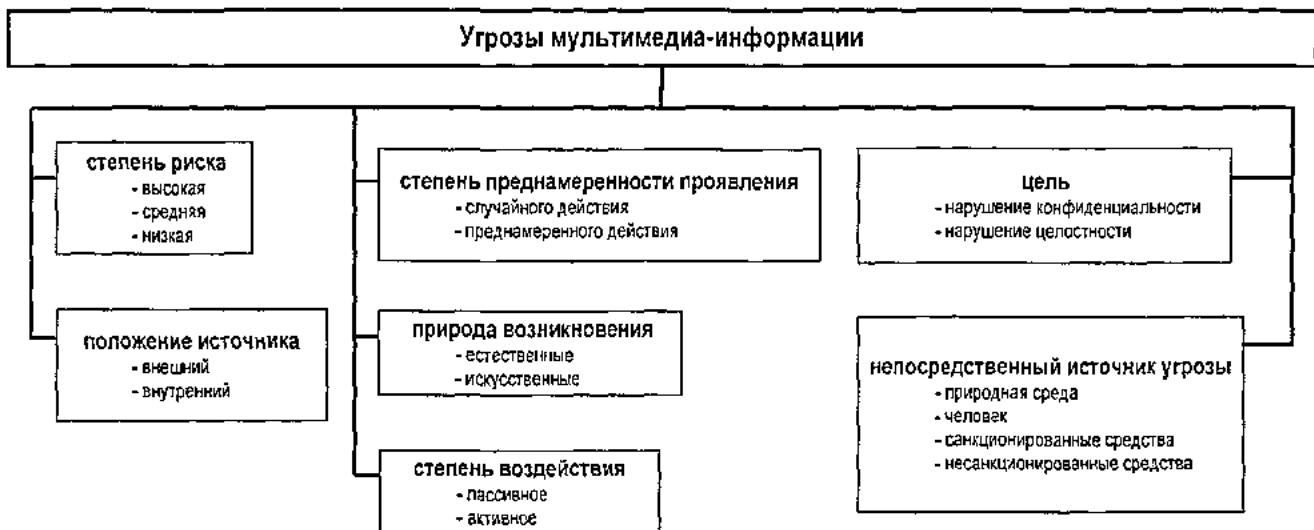


Рисунок 1 – Основные классификационные признаки угроз мультимедиа-информации

1.2. Исследование и анализ существующих методов внедрения цифровых водяных знаков

Данный параграф посвящен обзору и анализу существующих методов внедрения цифровых водяных знаков. В нем приведены современные термины и модели внедрения информации, описаны результаты анализа запатентованных решений в области внедрения цифровых водяных знаков и сформулированы основные недостатки данных методов. Рассмотрены методы оценки стойкости систем внедрения информации, необходимые для разработки методов внедрения и их последующего анализа.

Говоря о научной области исследований, невозможно обойтись без соответствующей терминологии и классификации основных понятий и объектов, которые изучаются данной наукой [62]. Основоположником современной, компьютерной стеганографии можно считать американского математика Густава Симмонса, который первым стал использовать математические модели для доказательства стеганографических задач [41, 40, 39]. В 1996 г. на первом открытом симпозиуме в Кембридже, официально посвященном проблематике сокрытия данных, были подведены итоги терминологическим дискуссиям [23]. Обсуждением руководил сподвижник стеганографии в Массачусетском Институте Технологии Росс Андерсон (Ross Anderson), весьма известный своими научными результатами и в криптографии. Результат обсуждения был зафиксирован Биргитом Пфитцманом. С этого момента можно считать зафиксированный в трудах симпозиума набор терминов устоявшимся, по крайней мере, в западной литературе.

Русская стеганографическая терминология на данный момент не имеет подобного кембриджскому документа. Учитывая, что после появления большого количества разнородных открытых материалов по криптографии, в криптографическую терминологию было внесено много путаницы, и она еще долго будет испытывать трудности из-за неточного определения многих терминов, к вопросу о стеганографической терминологии следует подходить чрезвычайно деликатно. Наиболее яркими представителями отечественной "стеганографической" школы являются Барсуков В.С. [88], Дворянкин С.В. [103], Романцов А.П. [104], Грибунин В.Г. [91, 102]. Однако анализ их научных работ [89, 87, 86, 101, 113] не позволяет сделать выводы о наличии единой устоявшейся системы терминов.

1.2.1. Анализ существующих методов внедрения цифровых водяных знаков

Проведенный анализ основан на исследовании существующих стеганографических программных средств и запатентованных решений в области стеганографии (см. Приложение А) [52, 54].

Основными стеганографическими понятиями являются *сообщение* и *контейнер*. Сообщением m называют секретную информацию, наличие которой необходимо скрыть. Всевозможные сообщения объединяются в пространство сообщений M э m .

Контейнером $b \in B$ называют несекретную информацию, которую используют для скрытия сообщений, где B – множество всех контейнеров. Секретный ключ k представляет собой некоторую секретную информацию, известную только законному пользователю. Через K обозначается множество всех допустимых секретных ключей. В общем случае в качестве сообщений, контейнеров и ключей могут быть использованы объекты произвольной природы. В наиболее развивающейся в последнее время *компьютерной стеганографии* в качестве сообщений, контейнеров и секретных ключей используют двоичные последовательности, т.е. $M = Z_2^n$ для некоторого фиксированного целого n , $B = Z_2^q$ и $K = Z_2^p$, при этом $q > n$ [92].

Пустой контейнер (или, еще говорят, *немодифицированный контейнер*) – это некоторый контейнер b , не содержащий сообщения. *Заполненный контейнер* (или соответственно *модифицированный контейнер*) – это контейнер b , содержащий сообщение m , в дальнейшем b_m .

Стеганографическим преобразованием над ними принято называть зависимости $F: M \times B \times K \rightarrow B$ и $F^{-1}: B \times K \rightarrow M$, сопоставляющие тройке (сообщение, пустой контейнер, ключ) контейнер результат и паре (заполненный контейнер, ключ) - исходное сообщение, т.е.

$$F(m, b, k) = b_{m,k}, F^{-1}(b_{m,k}) = m, \text{ где } m \in M; b, b_{m,k} \in B; k \in K.$$

Стеганографической системой называют (F, F^{-1}, M, B, K) совокупность сообщений, секретных ключей, контейнеров и связывающих их преобразований. Отметим, что приведено определение стеганографической системы принято считать современным. Существует более ранняя – классическая схема [14], являющаяся частным случаем данной схемы. Ее отличительной особенностью является отсутствие зависимости от секретного ключа, т.е.:

$$F(m, b) = b_m, F^{-1}(b_m) = m, \text{ где } m \in M; b, b_m \in B.$$

Как видно, терминология и математические модели стеганографии во многом сходны с понятиями и моделями, знакомыми нам из криптографии, однако они имеют иную смысловую окраску. В стеганографических системах часто используются методы криптографического преобразования данных [106, 111], поэтому важно верно определять смысловую нагрузку соответствующих терминов исходя из контекста. Одним из таких “скользких” терминов является понятие секретной компоненты системы - ключа, который является параметром как стеганографических, так и криптографических систем. Однако в разных системах он имеет различный смысл. Поэтому в случае, когда в стеганографической системе используются криптографические преобразования важно различать разные ключи, например, по их предназначению и способу использования.

Все существующие стеганографические системы можно разбить согласно их практическому предназначению на три тесно связанных между собой класса [78, 110, 64]:

- системы для скрытия секретных данных – используются для организации скрытых каналов передачи секретных данных. Основными требованиями, которые предъявляются к системам данного класса, являются невозможность обнаружения [125, 96], устойчивость к уничтожению и подмене передаваемых данных;

- системы для встраивания цифровых водяных знаков – предназначены для внедрения цифровых водяных знаков, которые используются для обеспечения защиты авторских или имущественных прав на цифровую информацию. Основными требованиями, которые предъявляются к системам данного класса, являются устойчивость кискажениям, уничтожению и подмене внедренных данных [117, 95, 90, 98];

- системы скрытой идентификации – служат для обеспечения идентификации, в частности, для маркирования изображений в электронных хранилищах цифровых изображений, аудио- и видеофайлов. В системах данного класса стеганографические методы используются не только для внедрения идентифицирующего заголовка, но и иных индивидуальных признаков файла (их иногда называют цифровыми отпечатками пальцев). Основным требованием, которое предъявляется к системам данного класса, является то, что изменения, происходящие в результате внедрения маркировочной информации, должны быть минимальными и не сказываться на качестве модифицированного контейнера [114].

В компьютерной стеганографии в качестве контейнеров могут быть использованы различные оцифрованные данные: растровые графические изображения, цифровой звук,

цифровое видео, всевозможные форматы хранения и передачи цифровой информации, текстовые и прочие электронные документы [8, 112–128]. Отметим, что в терминологии стеганографии существует два основных типа контейнеров: поточный и фиксированный. В случае, когда контейнер представляет собой объект с характеристиками конечных величин, его относят к фиксированному типу, в противном случае – к поточному.

Наиболее распространеными типами контейнеров в компьютерной стеганографии на данный момент являются изображения и аудиоданные, представленные в цифровой форме, а также видеопоследовательности, т.е. мультимедиа-контейнеры [123]. Это объясняется тем, что подобные контейнеры уже по технологии получения имеют шумовую составляющую, которая маскирует встраиваемое сообщение [115].

Довольно большой процент современных систем компьютерной стеганографии использует в качестве контейнеров растровые графические изображения различных форматов. Самое широкое распространение в последнее время получил формат JPEG. Практически все современные цифровые фотоаппараты и видеокамеры сохраняют изображения в этом формате, большинство фотографических изображений опубликованы в сети Интернет именно в нем. Есть и другие широко распространенные графические форматы, такие как Windows BMP, GIF, PNG, TIFF, TGA и т.д., но их рассмотрение выходит за рамки данного исследования.

С точки зрения разработки стеганографических систем важны следующие особенности современных графических форматов:

- наличие в формате сжатия данных;
- наличие в формате сжатия данных с потерями;
- использование в формате палитры цветов.

В том случае, когда формат хранения растровых изображений использует сжатие данных, значительно возрастает сложность разработки стеганографической системы, так как, во-первых, увеличивается сложность анализа формата, а во-вторых, вносимые стеганографической системой в данные изображения изменения приводят к нежелательному ухудшению эффективности сжатия. В случае, когда формат графических изображений использует сжатие с потерями информации, классические методы скрытия в графических изображениях, как правило, становятся малоэффективными, так как при потерях информации происходит уничтожение скрытой информации (в силу малой амплитуды скрытого сигнала). Поэтому в данном случае необходимо проводить

детальный анализ, направленный на поиск этапа сжатия, пригодного для сокрытия. После чего необходима адаптация классических методов сокрытия или разработка новых.

Использование в формате хранения графических изображений палитры цветов, также затрудняет применение классических методов сокрытия. К примеру, применим метод сокрытия в младших битах. Но не в младших битах данных графического изображения, которые являются ссылками на элементы палитры, а в младших битах элементов самой палитры, размер которой не превышает 256 цветов. При этом для сокрытия в младшие биты палитры необходимо, чтобы элементы палитры, отличающиеся лишь младшим битом, были близки по интенсивностям цветовых составляющих, что выполнено далеко не всегда (в связи с этим в подобных методах сначала выполняется этап преобразования палитры к требуемому виду).

Все методы, предназначенные для сокрытия данных, можно разделить по принципам, лежащим в их основе, на форматные и неформатные.

Форматные методы сокрытия (форматные стеганографические системы) — это такие методы (системы), которые основываются на особенностях формата хранения графических данных.

Разработка таких методов сводится к анализу формата с целью поиска служебных полей формата, изменение которых в конкретных условиях не скажется на работе с графическим изображением. Например, для сокрытия можно использовать те служебные поля формата, которые присутствуют в графических файлах, но не используются в настоящее время. Однако все форматные методы обладают общим недостатком — для них возможно построение полностью автоматического алгоритма, направленного на обнаружение факта сокрытия (с учетом принципа общеизвестности стеганографической системы). Поэтому их стойкость к атакам пассивных противников крайне низка.

Неформатные методы — методы, использующие непосредственно сами данные, которыми изображение представлено в этом формате.

Применение неформатных методов неизбежно приводит к появлению искажений, вносимых стеганографической системой, однако при этом они являются более стойкими к атакам как пассивных, так и активных противников.

Отмеченные выше особенности графических форматов важны лишь в случае разработки неформатных методов сокрытия, в которых сокрытие информации производится в полях данных, а не в служебных полях формата.

Другим типом контейнеров являются аудиосигналы. Этому типу контейнеров в данное время уделяется гораздо меньшее внимание. По-видимому, данный факт объясняется как сложностью обработки, так и более низким (по сравнению с изображениями) коэффициентом использования контейнера. Однако, в некоторых ситуациях данный тип контейнера является единственным возможным. Кроме того, из-за слабой распространенности и отсутствия наработок по обнаружению вложений этот контейнер может использоваться в случаях, требующих повышенной скрытности передачи небольших сообщений, к примеру, таких, как ЦВЗ.

1.2.2. Исследование методов оценки стойкости систем внедрения информации

По аналогии с криптографией [65, 77, 15], сторону, пытающуюся раскрыть стеганографическую систему передачи информации и определить наличие сообщения, называют *стеганоаналитиком* или, по аналогии с английскими источниками, *стегоаналитиком* (что с точки зрения этимологии является менее предпочтительным, поскольку слово "стего" есть не что иное, как сленговое сокращение от полного названия "стегано").

Соответственно, попытку определить наличие сообщения и его смысл называют атакой на стеганографическую систему. Задача стеганоаналитика состоит в раскрытии стеганографической системы и определении внедренного сообщения. В отличие от криптографии, под *раскрытием* (взломом) стеганографической системы принято понимать нахождение такой ее конструктивной либо иной уязвимости, которая позволяет определить факт сокрытия информации в контейнере, и возможность доказать данное утверждение третьей стороне с высокой степенью достоверности [13]. Учитывая это, аналогично криптографическим атакам, атаки на стеганографические системы можно разделить на классы [53, 60]:

- атаки со знанием только модифицированного контейнера – аналог криптографической атаки со знанием шифртекста. Стеганоаналитик в этом случае обладает только модифицированным контейнером, по которому он пытается определить наличие скрытого сообщения. Данный вид стеганографических атак – базовый из всех, по которым оцениваются стеганосистемы;

- атаки со знанием немодифицированного контейнера возможны в случае, когда стеганоаналитик также обладает способностью узнавать, какой именно

немодифицированный контейнер был использован для сокрытия сообщения. Данная атака определяет возможность определения факта сокрытия сообщений в дальнейшем, в зависимости от наличия однажды перехваченного контейнера и раскрытоого сообщения;

- об *атаках с выбором сообщения* говорят, когда стеганоаналитик имеет возможность указывать, какие именно сообщения будут скрыты, но при этом не имеет возможности указать контейнер, который будет для этого использоваться. Стойкость к данной атаке характеризует стойкость системы к перехвату и отслеживанию сообщений, посланных с использованием одного и того же контейнера. Данный вид атак иногда также позволяет определить тип примененной стеганографической системы;

- *атаки с выбором контейнера*, аналогично предыдущим, позволяют определить стойкость стеганосистемы к раскрытию, в случае повторного использования одного и того же сообщения с различными контейнерами;

- *атаки по подмене и имитации* не призваны определить факт наличия сообщения или извлечь его, их применяют для модификации скрытой информации либо имитации такой передачи;

- *атаки по противодействию передаче информации* используют для уничтожения скрытой информации и снижения пропускной способности каналов скрытой передачи данных [85].

При проведении анализа стеганографической системы следует по аналогии с криптографией учитывать необходимость применения правила Керкгоффа, которое заключается в том, что стойкость секретной системы должна определяться только секретностью ключа.

Сам процесс стеганоанализа контейнера можно разделить на два различных по сути действия:

- определение наличия сообщения в контейнере;
- извлечение содержания скрытого сообщения.

Стеганоаналитик, перед которым стоит задача первого или второго типа, является *пассивным нарушителем*, поскольку он не модифицирует доступные ему для анализа данные. С другой стороны, стеганоаналитик, который может вносить изменения в передаваемые по каналу данные, носит название *активный нарушитель* (*активный противник*). Атаки по подмене, имитации и противодействию характерны только для

активного нарушителя, в то время как остальные виды атак присущи только пассивным нарушителям.

Рассмотрим общую схему стеганографической системы при наличии как пассивного, так и активного противника (см. рисунок 2) [58]. Согласно данной схеме, на передающей стороне сообщение скрывается в контейнере при помощи прямого стеганографического преобразования. Затем полученный модифицированный контейнер отправляется по открытым каналам связи принимающей стороне, где после его получения при помощи обратного стеганографического преобразования извлекается исходное сообщение.

Как было сказано выше, задача пассивного противника состоит в определении факта наличия в контейнере скрытых данных, при этом делается допущение о том, что он может перехватывать все посланные контейнеры и анализировать их как по отдельности, так и в совокупности. В случае, если пассивному противнику удалось верно определить факт наличия скрытого сообщения в контейнере, он может пытаться извлечь его с целью ознакомления с его содержанием. Однако, как правило, перед сокрытием сообщение шифруется, и даже, если противнику удастся извлечь сообщение, то для ознакомления с его содержимым будет необходимо его дешифровать.

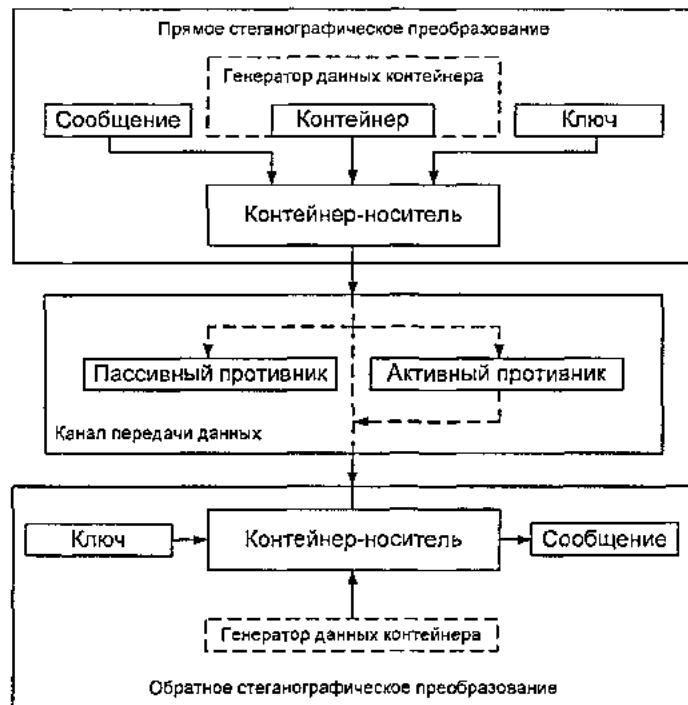


Рисунок 2 – Общая схема стеганографической системы, предназначеннной для передачи данных

Если же пассивный противник не сможет определить факт наличия сообщения, то попытки извлечь сообщение ни к чему не приведут в силу того, что эта задача будет иметь огромное множество решений, выбрать верное из которых практически невозможно.

Активный противник может вносить изменения в передаваемый по каналам связи контейнер [59], при этом предполагается, что ни на принимающей, ни на передающей стороне не знают, какой контейнер был изменен во время передачи, а какой – нет. На этот вопрос должно верно отвечать обратное стеганографическое преобразование. Самая простая задача активного противника – это уничтожение скрытой информации без определения факта наличия сообщения. Если же перед активным противником стоит задача изменения скрытого сообщения, то перед ее выполнением он должен проанализировать контейнер, чтобы удостовериться, что в контейнере действительно содержится сообщение, которое он должен изменить [80].

Очевидно, что стеганографическая система в первую очередь должна быть стойкой к попыткам определить факт ее использования. Но, если при этом она не будет стойкой к попыткам уничтожить скрытое с ее помощью сообщение, т.с. к попыткам понизить пропускную способность канала скрытой передачи (а в идеале — не допустить никакой скрытой передачи, даже той, наличие которой неизвестно), то эффективность ее практического использования при наличии активного противника может достигнуть нуля. Поэтому стойкость к таким атакам пассивного и активного противника – это и есть основное требование, которое предъявляется к стеганографической системе.

Впервые схема стеганографической системы для передачи данных с наличием только пассивного противника (см. рисунок 3) была подробно описана в работе Густава Симмонса в 1983 г. [41]. Представим себе, что Алиса и Боб находятся в тюрьме и хотят разработать план побега. Все их послания друг другу просматриваются противником (в данном случае это надзиратель Ева, которая желает разрушить их план и переведет их в тюрьму более строгого режима, как только поймет, что в их посланиях содержится скрытая от нее информация). Задача, стоящая перед Алисой и Бобом, состоит в том, чтобы разработать такой способ обмена информацией, при использовании которого Ева ничего не смогла бы заподозрить.

На рисунке 3 m — сообщение; B — пустой контейнер; B_m — модифицированный контейнер; k — секретный ключ, известный Алисе и Бобу. Алиса может посыпать как модифицированные, так и немодифицированные контейнеры.

Теоретическое обоснование стойкости стеганографических систем возможно при помощи, так называемой, *теоретико-информационной модели стойкости*. Для построения теоретико-информационной модели стойкости к атаке пассивного противника нам понадобятся некоторые понятия теории информации и способ проверки гипотезы. Условимся, что основание всех логарифмов в данном параграфе равно 2 и будем опускать его для краткости. Мощность множества S будем обозначать как $|S|$.

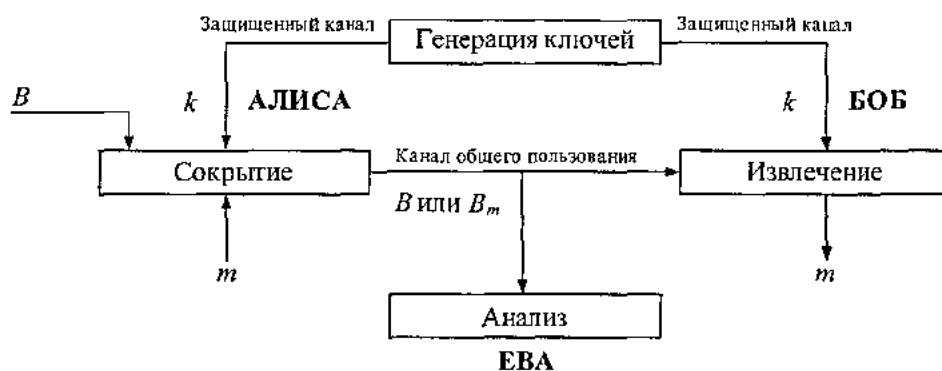


Рисунок 3 – Общая схема стеганографической системы
(при наличии пассивного противника)

Энтропия случайной величины X с распределением вероятностей P_X и алфавитом A_X определяется равенством:

$$H(X) = - \sum_{x \in A_X} P_X(x) \log P_X(x).$$

Условная энтропия случайной величины X по отношению к случайной величине Y есть

$$H(X | Y) = - \sum_{y \in A_Y} P_Y(y) H(X | Y = y),$$

где $H(X | Y = y)$ обозначает энтропию условной вероятности распределения $P(X | Y = y)$.

Проверка гипотезы состоит в том, чтобы определить, какая из двух гипотез H_0 или H_1 является верным толкованием наблюдаемой величины Q . Другими словами, есть два

возможных распределения вероятностей, которые принято обозначать P_{Q_0} и P_{Q_1} , над пространством \mathcal{G} возможных наблюдений. Если верна гипотеза H_0 , тогда Q была порождена согласно P_{Q_0} ; если же верна гипотеза H_1 , тогда Q была порождена согласно P_{Q_1} . Правило принятия решения – это двоичное отображение, заданное на пространстве \mathcal{G} , которое сопоставляет одну из двух возможных гипотез для каждого возможного элемента $q \in \mathcal{G}$. Две ошибки, которые могут быть совершены во время принятия решения, называются ошибками первого и второго рода. Ошибка первого рода состоит в том, что была принята гипотеза H_1 , тогда как верна гипотеза H_0 . Ошибка второго рода, состоит в том, что была принята гипотеза H_0 , тогда как верна гипотеза H_1 . Вероятность ошибки первого рода принято обозначать α , а второго – β .

Метод, позволяющий находить оптимальное решение, дает теорема Неймана-Пирсона. Правило принятия решения определяется путем сравнения с пороговым параметром T . В таком случае вероятности α и β являются функциями от T . Теорема гласит, что для любого параметра $T \in \mathcal{R}$ и максимально допустимой вероятности β ошибки второго рода вероятность ошибки первого рода α может быть минимизирована выбором гипотезы H_0 для наблюдаемой величины $q \in \mathcal{G}$ тогда и только тогда, когда:

$$\log \frac{P_{Q_0}(q)}{P_{Q_1}(q)} \geq T. \quad (1.1)$$

Левую часть этого выражения принято называть логарифмическим отношением правдоподобия. Как правило, для того, чтобы найти оптимальное решение, необходимо проверить достаточно много различных значений параметра T .

Основной мерой проверки гипотезы является относительная энтропия или различие между двумя распределениями вероятности P_{Q_0} и P_{Q_1} , определяемое следующим выражением:

$$D(P_{Q_0} \| P_{Q_1}) = \sum_{q \in \mathcal{G}} P_{Q_0}(q) \log \frac{P_{Q_0}(q)}{P_{Q_1}(q)}. \quad (1.2)$$

Относительная энтропия между двумя распределениями всегда неотрицательна, и равна нулю тогда и только тогда, когда распределения равны. Несмотря на то, что относительная энтропия не является метрикой с точки зрения математики (так как она не симметрична и не удовлетворяет аксиоме треугольника), полезно считать ее таковой.

Двоичная относительная энтропия $d(\alpha, \beta)$ определяется как:

$$d(\alpha, \beta) = \alpha \log \frac{\alpha}{1-\beta} + (1-\alpha) \log \frac{1-\alpha}{\beta}.$$

Энтропию, относительную энтропию и размер алфавита для любой случайной равномерно распределенной величины X на A^X , $X \in A^X$ связывает выражение

$$H(X) + D(P_X \| P_U) = \log |A_X|. \quad (1.3)$$

Относительная энтропия и проверка гипотезы связаны между собой теоремой Неймана-Пирсона: ожидаемое значение логарифмического отношения правдоподобия в (1.1) относительно P_{Q0} равно относительной энтропии $D(P_{Q0} \| P_{Q1})$ между P_{Q0} и P_{Q1} . Следующий стандартный результат показывает, что детерминированное отображение не может увеличить относительную энтропию между двумя распределениями.

Лемма 1. Пусть P_{Q0} и P_{Q1} — распределения вероятности на Θ . Тогда для любой функции $f: \Theta \rightarrow T$ такой, что $T_0 = f(Q_0)$, $T_1 = f(Q_1)$ выполнено неравенство:

$$D(P_{T_0} \| P_{T_1}) \leq D(P_{Q0} \| P_{Q1}).$$

Так как принятие решения между H_0 и H_1 — это частный случай детерминированного отображения, то вероятности ошибки первого рода α и ошибки второго рода β удовлетворяют неравенству:

$$d(\alpha, \beta) \leq D(P_{Q0} \| P_{Q1}). \quad (1.4)$$

Эта оценка обычно используется следующим образом. Предположим, что δ – это верхняя оценка для $D(P_{Q0} \parallel P_{Q1})$ и известна верхняя оценка для вероятности ошибки первого рода α , тогда с помощью (1.4) можно получить нижнюю оценку для вероятности ошибки второго рода β . Например, из того, что $\alpha = 0$, следует, что $\beta \geq 2 - \delta$.

Похожий результат имеет место и для обобщенного метода проверки гипотезы, где распределения P_{Q0} и P_{Q1} зависят от знания дополнительной случайной величины V , т.е. распределения вероятности принимают вид $P_{Q0|V=v}$ и $P_{Q1|V=v}$ для всех $v \in \Lambda$. Правило принятия решения в данном случае может зависеть от значения $V = v$, а вероятности ошибок примут вид $\alpha(v)$ и $\beta(v)$ для любого $v \in \Lambda$. Определим средние величины ошибок первого и второго рода соответственно как

$$\bar{\alpha} = \sum_{v \in \Lambda} P_V(v) \alpha(v),$$

$$\bar{\beta} = \sum_{v \in \Lambda} P_V(v) \beta(v).$$

Условную относительную энтропию между P_X и P_Y (над одним и тем же алфавитом, т.е. $A_X = A_Y$) зависящую от случайной величины Z определим как

$$D(P_{X|Z} \parallel P_{Y|Z}) = \sum_{z \in Z} P_Z(z) \sum_{x \in A_X} P_{X|Z=z}(x) \log \frac{P_{X|Z=z}(x)}{P_{Y|Z=z}(x)}. \quad (1.5)$$

Из неравенства Йенсена следует, что

$$d(\bar{\alpha}, \bar{\beta}) \leq D(P_{Q0|Z} \parallel P_{Q1|Z}). \quad (1.6)$$

Наша задача состоит в том, чтобы разработать модель процесса, позволяющую определять стойкость системы к атаке Евы. Не нарушая общности, будем полагать, что Алиса может действовать только в двух режимах: либо она активна (посыпает модифицированный контейнер B_m), либо пассивна (посыпает пустой контейнер B) [5].

Если Алиса активна, то она изменяет пустой контейнер B , скрывая в нем сообщение m с использованием секретного ключа k (отметим, что она также может

создавать контейнер B_m только лишь на основе сообщения m и ключа k). В результате сокрытия она получает модифицированный контейнер B_{mk} , который передает Бобу. Боб должен извлечь из контейнера B_{mk} скрытое сообщение m с помощью ключа k . Переходя к терминам энтропии, система должна удовлетворять следующим условиям:

$$H(B_m | B_{mk})=0. \text{ Условие однозначного определения контейнера входными данными.}$$

$H(m) > 0$. Условие невозможности определения факта наличия скрытого сообщения.

$$H(m | B_{mk}) = 0. \text{ Условие однозначного извлечения скрытого сообщения.}$$

Если Алиса пассивна, то она посыпает пустой контейнер B , и стеганографическая система не принимает участия в информационном обмене. В этом случае стеганографические преобразования, сообщение m и ключ k можно считать несуществующими.

Введенная таким образом модель не учитывает повторяемость данных, посланных Алисой Бобу. К примеру, если Алиса посыпает большое количество контейнеров Бобу, и хотя бы один из них содержит скрытую информацию, она считается активной, и B_m состоит из объединения всех посланных контейнеров.

Распределения вероятности предполагаются известными всем, если не оговорено противное. Отметим, что Боб (в отличие от Евы) знает, когда Алиса активна, а когда — нет.

Ева, анализируя контейнеры, посланные Алисой, должна определить какие из них относятся к распределениям пустого контейнера B , а какие — к распределениям модифицированного контейнера B_m , т.е. определить активность Алисы. Эта задача и есть проверка гипотезы. Определим стойкость стеганографической системы в терминах расстояния относительной энтропии между P_{B_m} и P_B .

Стеганографическая система, определенная выше, с пустым контейнером B и модифицированным контейнером B_m обладает стойкостью e к атакам пассивных противников, если:

$$D(P_B \| P_{B_m}) \leq e.$$

При этом, если $e = 0$, то стеганографическая система называется абсолютно стойкой к атакам пассивных противников.

Рассмотрим принятие решения Евой для частного случая принятия решения, определенного с помощью бинарного отношения (C_0, C_1) на множестве C' возможных контейнеров. Ева считает, что Алиса активна тогда и только тогда, когда анализируемый контейнер B содержится в C_1 . В лучшем случае, она всегда сможет верно определить факт сокрытия сообщения (но это возможно, только если Алиса выберет способ сокрытия, при котором пустые и модифицированные контейнеры будут отличаться). Если Ева не сможет определить, что она анализирует модифицированный контейнер, то она совершил ошибку второго рода (ее вероятность β).

Ошибка, которой обычно уделяют меньше внимания, – это ошибка первого рода: Ева считает, что Алиса послала модифицированный контейнер, но в действительности был послан пустой контейнер (вероятность ошибки первого рода α). Как частный случай, можно предположить, что в случае, когда Алиса пассивна, Ева не может сделать ошибку первого рода и обвинить ее в передаче скрытой информации (т.е. $\alpha = 0$).

Лемма 1 налагает ограничение на допустимые вероятности ошибок Евы. Из (1.4) легко получаем следующую теорему.

Теорема 1. В стеганографической системе со стойкостью e к атакам пассивных противников, вероятность β того, что противник не определит факт передачи скрытого сообщения, и вероятность α того, что противник ложно определит факт наличия скрытого сообщения, удовлетворяют неравенству:

$$d(\alpha, \beta) \leq e.$$

При этом, если $\alpha = 0$, то:

$$\beta \geq 2^{-e}.$$

Для абсолютно стойкой системы имеем $D(P_B \| P_{Bm}) = 0$ и, следовательно, $P_B = P_{Bm}$ (т.е. Ева не может с помощью анализа контейнера получить информацию об активности Алисы.)

Например, предположим, что у Алисы есть растровое графическое изображение I , которое она может послать Бобу. Используя модель восприятия, она определяет

множество M_I эквивалентных графических изображений, которые визуально неотличимы от I . Независимо от того, активна Алиса или нет, она посыпает случайно выбранный элемент M_I и это задает пространство вероятностей. Отметим, что в нашей модели противнику известно только множество M_I и, возможно, посланный контейнер I . Алиса может воспользоваться методами, описанными выше, для оценки и увеличения стойкости сокрытия сообщения. Однако, чтобы достичь стойкости против активных противников, которые могут вносить изменения в посланный контейнер, этих оценок недостаточно.

Можно предположить, что внешние факторы могут оказывать влияние на распределение вероятностей контейнера. Обозначим внешнюю, известную всем информацию через Y . Наша модель и определение стойкости стеганографической системы к атаке пассивного противника, приведенное выше, может быть адаптировано в соответствии с данным предположением. Для этого вычисляемые параметры должны зависеть от значения Y , и нам необходимо рассматривать средние вероятности ошибок первого и второго рода:

$$\bar{\alpha} = \sum_{y \in Y} P_Y(y) \alpha(y),$$

$$\bar{\beta} = \sum_{y \in Y} P_Y(y) \beta(y),$$

где $\alpha(y)$ и $\beta(y)$ — вероятности ошибок первого и второго рода для $Y = y$ соответственно.

Измененная стеганографическая система с внешней информацией Y , пустым контейнером B и модифицированным контейнером B_m обладает стойкостью e к атакам пассивных противников, если выполнено неравенство:

$$D(P_{B|Y} \| P_{B_m|Y}) \leq e.$$

Из (1.6) следует, что средние вероятности ошибок удовлетворяют неравенству из теоремы 1:

$$d(\bar{\alpha}, \bar{\beta}) \leq e.$$

С практической точки зрения, во всех видах атак стеганоаналитика интересует решение трех проблем: точное доказательство факта наличия сообщения в контейнере, определение его длины и нахождение смысла скрытого сообщения. До сих пор вторая из перечисленных задач остается мало исследованной, хотя она имеет множество немаловажных аспектов. Суть данной задачи заключается в определении для каждой стеганографической системы некоторого *порога обнаружения*, накладывающего ограничение на объем скрываемой информации. При превышении такого порога, т.е. при скрытии большего количества информации, чем это считается возможным, система становится уязвимой для стеганографических атак.

Очевидно, что чем меньше информации мы внедряем в контейнер, тем меньше вероятность обнаружения скрытого сообщения. При этом каждый стеганографический метод имеет свой собственный порог обнаружения, который позволяет скрывать различные объемы информации в различные контейнеры. Не следует путать порог обнаружения с *порогом скрытия* — максимальным объемом скрываемой информации стеганосистемы.

Не является вполне очевидным тот факт, что порог обнаружения стеганографической системы может зависеть не только от контейнера, но и от скрываемого сообщения. Чтобы показать это, достаточно построить абстрактную стеганографическую систему, обобщающую одну из наиболее известных. То же самое утверждение легко доказывается и на практике, когда дело доходит до применения адаптивных стеганографических систем. То же самое можно сказать и про порог скрытия. Такой очевидной теме посвящено чрезвычайно мало научных публикаций.

Пфитцман и Вестфелд [43] в 2000 г. разработали довольно мощную стеганографическую атаку, которая может быть применена к любой стеганосистеме, меняющей статистическое распределение элементов пространства скрытия. Для этого используется анализ гистограммы, полученной по элементам пространства и оценка пар значений (ПЗ) этой гистограммы. Пары значений могут быть сформированы значениями пикселей изображения, квантовыми коэффициентами дискретного косинусного преобразования или индексами палитры, которые отличаются только по младшему биту.

Перед внедрением в контейнере два элемента каждой ПЗ распределены неравномерно. После внедрения сообщения вероятности элементов в каждой паре

стремятся стать равными (даный факт, конечно, зависит от длины сообщения). Так как увеличение одного элемента ПЗ никак не меняет сумму значений обоих пикселей в изображении, можно использовать этот факт, чтобы спроектировать новый метод стеганоанализа на основе проверки статистической гипотезы. Для этого необходимо лишь проверить независимость распределений элементов каждой из ПЗ.

Если (в дополнение к этому) стеганосистема внедряет биты сообщения последовательно в следующие друг за другом пиксели, индексы или коэффициенты, начиная при этом в верхнем левом углу, проссывая все изображение, мы сможем проследить резкую смену значений выходных параметров нашего статистического теста.

Например, для метода внедрения в палитру изображения не может быть более 256 цветов c_i в палитре, т.е. не более 128 пар значений. Для каждой i -й пары, $i = 1, \dots, k$ мы определяем

$$n_i^1 = \frac{1}{2} (\text{количество индексов во множестве } \{c_{2i}, c_{2i+1}\}) \text{ и } n_i = c_{2i} n_i^1.$$

Тогда n_i^1 — ожидаемая частота; а n_i — действительная частота появления цвета c_{2i} . Учитывая данные определения, мы можем определить метод проверки статистической гипотезы с помощью теста χ^2 (хи-квадрат). Статистика χ^2 с k степенями свободы вычисляется следующим образом:

$$\chi^2_{k-1} = \sum_{i=1}^k \frac{(n_i - n_i^1)^2}{n_i^1},$$

где значение p определено как

$$p = 1 - \frac{1}{2^{\frac{k-1}{2}} \Gamma(\frac{k-1}{2})} \int_0^{\chi^2_{k-1}} e^{-\frac{x}{2}} x^{\frac{k-1}{2}-1} dx.$$

Для последовательно внедренного сообщения, можно просмотреть изображение в том же самом порядке, в котором сообщение было внедрено, и проследить изменение величины p для множества пройденных пикселей. Значение p сначала будет близко к 1, а затем внезапно упадет почти до 0 — в месте, где закончится скрытое сообщение. Нулевое

значение сохранится до нижнего правого угла изображение, чего конечно и следовало ожидать.

Если несущие сообщение отсчеты контейнера (в нашем случае пиксели в изображении) выбраны беспорядочно, а не последовательно, данный тест становится менее эффективным, если большинство отсчетов не использовалось при сокрытии.

1.3. Обзор и анализ существующих методов защиты и контроля целостности мультимедиа-информации

В данном параграфе описаны результаты анализа существующих симметричных криптографических преобразований и методов контроля целостности мультимедиа-информации. В нем проводится обзор и анализ криптографических преобразований, которые используются при построении систем защиты мультимедиа-информации, а именно симметричных шифров, бесключевых хэш-функций, хэш-функций с зависимостью от секретного ключа.

Проведенный в предыдущем параграфе анализ современных методов внедрения цифровых водяных знаков показал, что для обеспечения стойкости стеганографических систем к атакам активных и пассивных противников, необходимо использовать ряд криптографических инструментов. В частности, для защиты от подделки скрываемого ЦВЗ необходимо использовать его шифрование перед сокрытием. Кроме того, шифрование позволяет увеличить энтропию ЦВЗ, что значительно усложняет процесс стеганоанализа.

Основой современных систем защиты информации и контроля целостности также являются криптографические преобразования, такие как симметричные шифры, бесключевые хэш-функции, хэш-функции с зависимостью от секретного ключа. Поэтому в рамках данного диссертационного исследования важно провести их анализ, выявить их основные свойства, особенности практического использования, достоинства и недостатки.

1.3.1. Анализ архитектуры программных систем защиты и контроля целостности информации

На сегодняшний день существует огромное количество средств программной и программно-аппаратной защиты информации, решающих частные задачи обеспечения информационной защиты и конфиденциальности при работе с разнородной информацией.

Практически все из них имеют обобщенно одинаковую статичную архитектуру, детали которой зависят от выполняемой частной задачи, и функциональность, зависящую от требуемых от программно-аппаратного средства решений.

Так, средства контроля целостности информации состоят из следующих модулей:

- модуля контроля целостности;
- модуля хранения служебной информации (включающей коды контроля целостности);
- модуля журналирования;
- модуля управления комплексом

и имеют в общем случае следующую архитектуру (см. рисунок 4).

Модуль контроля целостности взаимодействует со средой хранения информации, либо осуществляя контроль целостности, в случае когда для информационной единицы есть вычисленный код контроля целостности, либо вычисляя код контроля целостности и добавляя информацию об информационной единице.

Модуль хранения служебной информации осуществляет хранение информации, необходимой для функционирования комплекса, включающей различные настройки, коды контроля целостности и т.д. Кроме того, в его функции входит обработка запросов, поиск и предоставление необходимой другим модулям информации.



Рисунок 4 - Обобщенная архитектура программной системы контроля целостности

Модуль журналирования обеспечивает сбор и хранение информации о функционировании системы, включающей сообщения о производительности, о выполненных операциях, о произошедших нарушениях и сбоях в работе системы. Данная

информация необходима администратору средства защиты информации для отслеживания состояния системы и коррекции ее настроек.

Модуль управления позволяет в оперативном режиме управлять действиями и изменять функциональность комплекса в соответствии с текущими запросами пользователей, меняющейся политикой безопасности и настройками системы.

Данный подход предоставляет возможность универсального подхода к вопросам защиты информации, однако его излишняя обобщенность приводит к серьезным просчетам и ошибкам проектирования при реализации представленной выше архитектуры.

Рассмотрим существующие системы защиты и контроля целостности, чтобы определить наиболее значимые характеристики, компоненты архитектуры современной системы защиты и выявить их недостатки, которые могут помешать достижению поставленной в исследовании цели.

1.3.2. Анализ методов защиты информации

Одним из основных способов безопасного хранения, использования и передачи конфиденциальной информации является применение к сохраняемым данным методов криптографического преобразования данных [73, 55]. Последнее время много внимания уделяется новым асимметричным крипtosистемам [10], которые позволяют строить защищенные каналы передачи данных без предварительной договоренности о секретной компоненте шифра - ключах. Тем не менее, они обладают рядом недостатков, которые отсутствуют у симметричных шифров [116]. К ним, например, можно отнести малую скорость зашифрования данных и долгое время выработки ключей шифрования, что делает их непригодными при обмене данными в режиме реального времени и сильно затрудняет их аппаратную реализацию [56].

Под *детерминированной системой шифрования* будем понимать отображение $F:P \times K \rightarrow C$, где P , C и K - непустые множества, называемые соответственно пространством открытых текстов или входов, пространством шифртекстов или выходов и пространством ключей, причем при каждом фиксированном $k \in K$ отображение $F_{(k)}:P \rightarrow C = F(p,k)$ является мономорфизмом. Последнее необходимо для возможности однозначного расшифрования шифртекста. Для краткости детерминированную систему шифрования принято называть просто *шифром*. Порядком системы называется число различных отображений $F_{(k)}$, то есть таких, что $F_{(k1)} \neq F_{(k2)}$ при $k1 \neq k2$. Степенью системы

принято называть мощность множества P . Шифр называется *нерастягивающим*, если $|P|=|C|$, *растягивающим*, если $|P|<|C|$ и *эндоморфным*, если $P=C$. Если разным ключам системы соответствуют различные отображения F , то система называется *точной*. Под *алфавитом* будем понимать конечное множество A , элементы которого называются *символами* (буквами, знаками). Тогда *текст* – это упорядоченный набор символов. Стоит отметить так же два важных свойства систем шифрования, которыми должны обладать все стойкие криптографические отображения. *Рассеиванием* называется влияние одного знака открытого текста или ключа на как можно большее количество знаков шифртекста. Это свойство делает восстановление неизвестного ключа по частям трудной, а в идеале неразрешимой задачей. Под *перемешиванием* понимают использование таких преобразований, которые усложняют поиск связи между открытим и зашифрованным текстом методами статистического анализа или делают его невозможным (трудно осуществимым).

Все симметричные системы шифрования можно характеризовать по способам преобразований, выполняемых с открытым текстом: перестановки, моноалфавитные замены, полиалфавитные замены, подстановки, композиционные и итерационные шифры, шифры Файстеля [108].

Шифры перестановки являются простейшими и, вероятно, самыми древними. Они заключаются в том, что символы открытого текста представляются по определенному правилу, зависящему от ключа, в пределах этого текста.

Моноалфавитными заменами называют шифры, в которых символы алфавита открытого текста меняются на символы того же алфавита по определенному правилу, зависящему от ключа.

Полиалфавитные замены – это шифры, в которых символы открытого текста меняются на символы того же, либо другого алфавита по определенному правилу, зависящему от ключа и от положения символа в тексте. Полиалфавитная замена называется *подстановкой*, если алфавиты открытого и зашифрованного текстов совпадают.

Пусть имеется семейство систем шифрования $F_i: P \times K_i \rightarrow C$, $i=1\dots n$, для которых $P=C$, то есть пространства входов и выходов всех шифров совпадают. Тогда *композиционным (каскадным, послойным) шифром* будет называться шифр, полученный с помощью композиции шифров семейства, задаваемый отображением $F: P \times (K_1 \times \dots \times K_n) \rightarrow P$,

где $F=F_n \circ F_{n-1} \circ \dots \circ F_1$. Его ключом будет $k=(k_1, \dots, k_n) \in K_1 \times \dots \times K_n$. При этом отображение F_i принято называть *i-м циклом шифрования*, K_i – *ключом i-го цикла шифрования* или *цикловым ключом*.

Итерационным шифром называется композиционный шифр, у которого совпадают все циклы шифрования F_i и ключевые пространства K_i . Говоря неформально, итерационный шифр – это применение одной и той же криптографической функции n раз с вообще говоря разными ключами.

Шифром Файстеля (сбалансированной сетью Файстеля) называется композиционный шифр $F=F_n \circ F_{n-1} \circ \dots \circ F_1$, если $P=T \times T$, то есть $\forall p \in P \exists l, r \in T$ такие, что $p=(l,r)$, и $F_i(l,r)=(r, l \otimes f_i(r))$, где f_i – отображение, зависящее от циклового ключа k_i , \otimes – это любая операция, относительно которой замкнуто множество T . Если в качестве T выбрать векторное пространство Z_2^m , а в качестве \otimes векторное сложение в этом пространстве, то при случайному выборе цикловых ключей и достаточно большом n такой шифр Файстеля будет представлять псевдослучайную подстановку на Z_2^{2m} , имеющие характеристики весьма близкие к случайной подстановке.

Пусть $F=F_1 \circ F_2 \circ \dots \circ F_n: P \times (K_1 \times \dots \times K_n) \rightarrow P$ – произвольный композиционный шифр, и на множестве P задана произвольная групповая операция \otimes . Для $\forall p, p' \in P$ определим $\Delta p=p \otimes (p')^{-1}$, где $(p')^{-1}$ – это элемент обратный к p' относительно операции \otimes . Пусть $p_i=F_i(k_i, p_{i+1})$, $p'_i=F_i(k_i, p'_{i+1})$, $\Delta p_i=p_i \otimes (p'_i)^{-1}$, $i=1, 2, 3, \dots, n$. Тогда композиционную систему шифрования F будем называть *марковской*, если последовательность $\{\Delta p_i\}$ образует цепь Маркова.

Помимо этого, принято делить все симметричные системы на две группы: блочные и поточные. Шифр называют *блочным*, если шифрование каждого блока открытого текста не зависит от других блоков, откуда следует, что результаты шифрования двух одинаковых блоков одного и того же открытого текста совпадают. Если же результат шифрования очередного блока открытого текста зависит не только от него самого и секретного ключа, но и в общем случае от предыдущих блоков, то этот шифр называют *поточным*. Существует более мягкое "практическое" определение: блочные шифры те, в которых текст делится на порции по несколько октетов, поточные же системы оперируют данными по одному биту или символу.

Представляет интерес следующее формальное определение системы блочного шифрования: шифр $F: P \times K \rightarrow C$ называется *блочным*, если множества P, K, C – конечны. В

поточном же шифровании, это требование нарушается, к примеру, если во время начала шифрования, вообще говоря, не известно количество символов в тексте, то удобно считать, что множество P не является конечным. Отметим, что в большинстве современных поточных систем результат шифрования текущего блока открытого текста зависит от его номера и не зависит от самих предыдущих блоков. Такие симметричные криптосистемы называют *шифрами гаммирования*. В этом случае стойкость системы определяется исключительно свойствами гаммы (являющейся обычно псевдослучайной числовой последовательностью).

Введем понятие теоретической стойкости шифра. Пусть имеем $F:P \times K \rightarrow C$, $k \in K$ – произвольный фиксированный ключ, P состоит из n открытых текстов p_i , $i=1 \dots n$; v_i , $i=1 \dots n$ – вероятность того, что i -ый текст будет зашифрован (вероятность появления i -го текста). Тогда под *априорной мерой неопределенности открытого текста* будем понимать,

$$H(P) = -(v_1 \log_2 v_1 + v_2 \log_2 v_2 + \dots + v_n \log_2 v_n).$$

В случае когда о P нет никакой априорной информации, все тексты считаются равновероятными и $H(P) = |P|$, где через $|P|$ обозначается размер текстов в битах.

Пусть имеем $c \in C$ – некий определенный шифртекст; w_i , $i=1 \dots n$ – вероятность того, что $F(p_i, k) = c$ верное равенство. Тогда под *апостериорной мерой неопределенности открытого текста* будем понимать

$$H(P|C) = -(w_1 \log_2 w_1 + w_2 \log_2 w_2 + \dots + w_n \log_2 w_n).$$

Тогда $I = H(P) - H(P|C)$ определяет количество информации об открытом тексте в битах, которую можно извлечь из шифртекста. Шифры, для которых выполнено $H(P) = H(P|C)$ принято называть *абсолютно стойкими (совершенными)*.

Пусть K состоит из m ключей k_i ; v_i , $i=1 \dots m$ – вероятность использования i -го ключа. Тогда под *мерой неопределенности криптосистемы* (*мерой неопределенности секретного ключа*) будем понимать

$$H(K) = -(v_1 \log_2 v_1 + v_2 \log_2 v_2 + \dots + v_m \log_2 v_m).$$

Отметим, что в случае равновероятного распределения элементов в пространстве ключей $H(K) = |K|$. Шеннон предложил и доказал необходимое условие абсолютной стойкости шифра, которое заключается в том, что

для того, чтобы шифр был абсолютно стойким, необходимо, чтобы неопределенность системы шифрования была не меньше неопределенности открытого текста: $H(K) \geq H(P)$.

Из этого условия и правила Кирхгофа следует что, для того, чтобы шифр был абсолютно стойким, необходимо, чтобы размер использованного для шифрования ключа был не меньше размера шифруемых открытых текстов: $|K| \geq |P|$. Равенство возможно, если $H(K) = |K|$, то есть если все ключи равновероятны [61, 68].

Шифры не являющиеся абсолютно стойкими принято называть *несовершенными*.

Определим функцию *ненадежности ключа* как неопределенность ключа при известных n битах шифртекста. Расстоянием единственности шифра назовем минимальное количество бит, при котором функция ненадежности близка к нулю. Шеннон показал, что эти величины зависят от избыточности открытого текста, причем расстояние единственности прямо пропорционально размеру ключа и обратно пропорционально избыточности $R = 1 - H(P)/|P|$. Следовательно, при $R=0$ невозможен криптоанализ со знанием только шифртекста даже при условии неограниченных вычислительных ресурсов.

1.3.3. Анализ современных методов контроля целостности

Функцией хэширования (хэш-функцией) называют функцию h , удовлетворяющую как минимум двум требованиям [66]:

- *сжатие* – функция h отображает входное сообщение x произвольной конечной длины в хэш-значение $y = h(x)$ небольшой фиксированной длины (далее, входное сообщение будем называть прообразом);

- *простота вычисления* – для заданной функции h и сообщения x , $h(x)$ вычисляется не выше, чем с полиномиальной сложностью.

Отметим, что функции хэширования, используемые в криптографических приложениях, должны удовлетворять дополнительным требованиям. Все существующие хэш-функции можно разделить на два больших класса: бесключевые хэш-функции, зависящие только от сообщения, и хэш-функции с секретным ключом, зависящие как от сообщения, так и от секретного ключа.

По используемым внутренним преобразованиям функции хэширования можно разделить на:

- функции, использующие битовые логические преобразования. Эти функции применяют к входному сообщению побитовые нелинейные операции "И", "ИЛИ", "НЕ", "ИСКЛЮЧАЮЩЕЕ ИЛИ", различные сдвиги и т.п.;

- функции, использующие блочные симметричные шифры;

- функции, использующие преобразования в группах, полях и кольцах с целочисленным или полиномиальным базисом;
- функции, использующие матричные преобразования.

Приведем основные требования, которые предъявляются к используемым в криптографии бесключевым хэш-функциям:

- *стойкость к вычислению прообраза* – невозможность нахождения неизвестного прообраза для любых предварительно заданных хэш-значений, т.е. для заданной хэш-функции h вычислительно невозможно найти неизвестный прообраз x при предварительно заданном хэш-значении $y = h(x)$ для любого значения y . Под термином "вычислительно невозможно" здесь и далее будем понимать, что алгоритм, выполняющий данное преобразование, обладает не менее чем экспоненциальной сложностью.

- *стойкость к вычислению второго прообраза* – невозможность нахождения любого другого прообраза, который давал бы такое же хэш-значение, как и заданное, т.е. для заданной хэш-функции h и прообраза x вычислительно невозможно найти другой прообраз $x' \neq x$, для которого выполнялось бы условие $h(x) = h(x')$.

- *стойкость к коллизиям* – невозможность нахождения двух прообразов для которых вырабатывалось бы одинаковое значение, т.е. для заданной хэш-функции h вычислительно невозможно найти два прообраза x и x' , $x \neq x'$, для которых выполнялось бы условие $h(x) = h(x')$.

Отметим, что требование стойкости к коллизиям является более жестким, чем требование стойкости к вычислению второго прообраза, так как предполагает произвольный выбор двух прообразов.

Однонаправленной хэш-функцией называется функция h , удовлетворяющая требованиям сжатия, простоты вычисления, стойкости к вычислению прообраза и стойкости к вычислению второго прообраза.

Бесколлизионной хэш-функцией называется функция h , удовлетворяющая требованиям сжатия, простоты вычисления, стойкости к вычислению второго прообраза и стойкости к коллизиям.

На практике предпочтительнее использовать хэш-функции, являющиеся одновременно бесколлизионными и однонаправленными.

Основным дополнительным требованием, которое предъявляется к функциям с секретным ключом является *вычислительная стойкость*, т.е. невозможность нахождения

хэш-значения для заданного сообщения без известного секретного ключа. Говоря иначе, для заданной ключевой хэш-функции h и одной или более корректных пар прообразов и хэш-значений $(x_i, h(x_i, k))$ при неизвестном секретном ключе k вычислительно невозможно найти другую корректную пару $(x, h(x, k))$ для любого $x \neq x_i$.

Требование вычислительной стойкости предполагает выполнение требования стойкости ключа (по одной или более корректных пар прообразов и хэш-значениям $(x_i, h(x_i, k))$ вычислительно невозможно восстановить секретный ключ k), однако, требование стойкости ключа не предполагает выполнение требования вычислительной стойкости.

Наиболее распространенными хэш-функциями, используемыми в настоящее время в криптографических системах защиты информации, являются: ГОСТ 34.11-94, HAVAL, SHA-1, RIPEMD-160, MD4, MD5, ГОСТ 28147-89 режим 4, Rijndael CBC-MAC, Whirlpool, SHA-2, UMAC.

Некоторые характеристики современных хэш-функций представлены в таблице 1.

Таблица 1 – Характеристики современных хэш-функций

Функция хэширования	Базовые преобразования	Длина хэш-значения, бит
Whirlpool	в конечных полях и матрицах	512
SHA-2	логические и арифметические	256, 384, 512
ГОСТ 34.11-94	блочный симметричный шифр	256
HAVAL	логические и арифметические	128, 160, 192, 256
SHA-1	логические и арифметические	160
RIPEMD-160	логические и арифметические	160
MD5	логические и арифметические	128
MD4	логические и арифметические	128
UMAC	в колышах	128, 64
Rijndael CBC-MAC	блочный симметричный шифр	128
ГОСТ 28147-89 (режим 4)	блочный симметричный шифр	64

В качестве основных критериев оценки функций хэширования можно использовать стойкость и вычислительную сложность вычисления значения хэш-функции. Минимальные требования по стойкости, определенные при разработке стандарта AES, соответствуют сложности атаки, равной 2^{128} . Таким образом, для разрабатываемых криптографических систем можно рекомендовать только односторонние хэш-функции, имеющие стойкость к коллизиям не менее 2^{128} , т.е. функции с длиной хэш-значения не менее 256 бит. Односторонние функции хэширования с длиной хэш-значения, равного 128 битам, следует по возможности исключать из применения.

Для практического использования хэш-функций с секретным ключом, необходимо чтобы длина ключа и хэш-значения составляла не менее 128 бит, однако, в настоящее время возможно использование функций с длиной ключа не менее 128 бит и длиной хэш-значения не менее 64-х бит [69].

1.4. Постановка задачи исследования

В ходе проведенного исследования были изучены угрозы безопасности мультимедиа-информации, существующие методы внедрения цифровых водяных знаков в различные форматы хранения мультимедиа-информации, симметричные криптографические преобразования и методы контроля целостности.

Увеличение количества публикуемых работ и патентуемых решений, связанных с вопросами стеганографии, несомненно, указывает на увеличение темпов развития данной науки. Однако, анализ запатентованных решений в области внедрения цифровых водяных знаков, использующих эти достижения показывает, что существующие на сегодняшний день методы имеют ряд существенных недостатков. В частности, они не могут полностью удовлетворить существующим на сегодняшний день угрозам безопасности мультимедиа-информации и, как следствие, требованиям существующих практических задач. Распространенными недостатками существующих систем является низкая скрытность и невысокая надежность используемых в них методов сокрытия.

Кроме того, на сегодняшний день не существует системы внедрения цифровых водяных знаков, которая могла бы решить одновременно задачу контроля целостности графических изображений, задачу контроля за различными копиями одного и того же изображения и задачу классификации различных изображений в хранилищах мультимедийных данных.

Очевидна необходимость проведения разработки системы внедрения идентификационных данных и кода контроля целостности в растровые графические изображения, которая должна удовлетворять следующим требованиям:

- минимизация искажений, вносимых при сокрытии - система не должна оказывать влияния на перцептивные свойства мультимедиа-контейнеров;
- невозможность определения факта наличия сокрытых в контейнере данных - возникает из практических соображений, делающих невозможным определение

неправомочными пользователями факта наличия скрытой информации с целью недопустить доступа к ней;

- невозможность извлечения сокрытых данных при условии известного модифицированного контейнера и неизвестном секретном ключе - возникает из практических соображений, делающих невозможным ознакомление неправомочных пользователей с содержанием сообщения,

- устойчивость к фальсификации сокрытых данных.

В связи с этим, целью данного исследования становится разработка программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров.

И в соответствии с поставленной целью, в работе необходимо решить следующие научные задачи:

- разработать модель естественности мультимедиа-информации, предназначенную для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков. Данная модель должна позволять численно оценивать статистические изменения, характеризующие возможность определения факта наличия сокрытого водяного знака;

- разработать методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров. Данные методы должны строиться в соответствии с разработанной моделью естественности. Метод сокрытия должен зависеть от секретного ключа, однозначно определяющего способ внедрения и извлечения информации;

- разоработать метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков. Данный метод должен позволять использовать существующие хэш-функции и позволять проводить контроль целостности только при наличии мультимедиа-контейнера, секретного ключа и возможно ключа хэш-функции;

- разработать модель оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки. Подобная модель необходима для получения разносторонних численных характеристик, позволяющих оценить ухудшения качества мультимедиа-информации;

- разработать обобщенную архитектуру программного комплекса контроля целостности мультимедиа-информации, включающую интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения программного обеспечения стороннего разработчика. Данная архитектура должна позволять осуществлять масштабирование комплекса и его настройку в соответствии с уже имеющейся информационно-вычислительной инфраструктурой. Открытые интерфейсы должны позволять производить изменение и наращивание функциональности за счет подключения модулей сторонних разработчиков.

1.5. Выводы

1. В данной главе были проанализированы существующие угрозы безопасности мультимедиа-информации, рассмотрены первичные классы угроз. В результате анализа были выявлены основные угрозы безопасности мультимедиа-информации: нарушение конфиденциальности и нарушение целостности. Были исследованы средства и методы внедрения цифровых водяных знаков, рассмотрены их основные свойства. Рассмотрены основные типы атак на системы внедрения цифровых водяных знаков и методы оценки стойкости систем внедрения информации, необходимые для разработки методов внедрения и их последующего анализа. Проанализированы криптографические преобразования, которые являются основой систем защиты мультимедиа-информации и контроля ее целостности, а именно симметричные шифры, бесключевые хэш-функции, хэш-функции с зависимостью от секретного ключа.

2. В процессе проведения исследования были выявлены недостатки, присущие всем перечисленным методам и системам. А именно, существующие системы внедрения водяных знаков обладают низкой стойкостью к атакам активных и пассивных противников, при этом в них отсутствуют методы анализа нарушения перцептивного качества и статистических свойств мультимедиа-контейнеров, затрудняющие принятие решения о возможности их использования на практике. Существующие системы контроля целостности основаны на использовании хэш-функций, недостатком такого подхода является необходимость организации хранения вычисленных хэш-значений и установления связи между объектом контроля и хэш-значением.

3. С учетом выявленных недостатков и актуальности выбранной темы исследования была поставлена цель диссертационной работы, которая заключается в разработке программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров.

4. В соответствии с поставленной целью были сформулированы следующие научные задачи исследования:

- разработать модель естественности мультимедиа-информации, предназначенную для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков;
- разработать методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров;
- разработать метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков;
- разработать модель оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки;
- разработать обобщенную архитектуру программного комплекса контроля целостности мультимедиа-информации, включающую интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения программного обеспечения стороннего разработчика.

Глава 2. Разработка математических моделей и методов контроля целостности мультимедиа-информации путем скрытого внедрения и извлечения цифровых водяных знаков

В предыдущей главе был сделан обзор методов внедрения цифровых водяных знаков. Так же был проведен анализ методов и алгоритмов защиты мультимедиа-информации и методов контроля целостности. Выделены недостатки существующих методов, в качестве основного недостатка была отмечена низкая стойкость существующих систем внедрения информации.

Данная глава посвящена построению математического аппарата контроля целостности мультимедиа-информации путем скрытого внедрения и извлечения цифровых водяных знаков. Т.е. в данной главе будут рассмотрены следующие частные задачи диссертационного исследования:

- разработка модели естественности мультимедиа-информации, предназначенной для статистической оценки изменений мультимедиа-контейнеров, вызванных внедрением цифровых водяных знаков. Отличительной особенностью разрабатываемой модели естественности является понятие пространства сокрытия в мультимедиа информации, которое позволяет использовать модель для различных видов и форматов хранения мультимедиа-информации;

- разработка методов внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров. Данные методы зависят от секретного ключа, однозначно определяющего способ внедрения и извлечения информации и позволяют равномерно распределять внедряемую информацию по пространству сокрытия;

- разработка метода контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков. Данный метод в корне отличается от существующих тем, что при помощи использования стандартных хэш-функций он позволяет проводить контроль целостности при наличии только мультимедиа-контейнера и секретного ключа;

- разработка модели оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки. Данная модель необходима для получения разносторонних численных характеристик, позволяющих оценить ухудшения качества мультимедиа-информации.

2.1. Разработка модели естественности мультимедиа-информации

Под мультимедиа- информацией принято понимать цифровой звук, цифровые графические изображения и цифровое видео. Не нарушая общности, для простоты изложения в дальнейшем ограничимся рассмотрением цифрового звука и растровых графических изображений.

Модель естественности мультимедиа информации предназначена для статистической оценки искажений вносимых при внедрении цифрового водяного знака в мультимедиа-контейнер [76]. Основой разработанной модели естественности является понятие пространства сокрытия, которое определено в следующем параграфе.

2.1.1. Определение пространства сокрытия в мультимедиа-информации

Цифровые аудио-сигналы представляют собой квантованные по уровню дискретные сигналы и описываются квантованными решетчатыми функциями. Число уровней квантования N и наименьшее число разрядов m двоичных чисел, кодирующих эти уровни, связаны соотношением $m=[\log_2 N]$. Здесь под $[\cdot]$ понимается операция взятия наименьшего целого числа не меньше числа $\log_2 N$.

К настоящему времени разработано достаточно большое число алгоритмов, предназначенных для уменьшения ширины полосы, необходимой для передачи цифровых аудио данных. Некоторые из разработанных алгоритмов позволяют уменьшить требования к ширине полосы со 128 кБит/сек. до 1200 Бит/сек. Несмотря на большое разнообразие, большинство вокодеров строятся по сходным принципам. Число различных же схем построения вокодеров не столь велико и составляет немногим более десятка.

Одними из первых появились вокодеры, представляющие сигнал во временной области. Данные вокодеры получили название кодеры формы. К данной группе относятся вокодеры, описываемые спецификациями G.711 (aLaw и uLaw), G.721 (со скоростью передачи 32 кБит/сек.), G.723 (со скоростями передачи 16, 24 и 40 кБит/сек.).

К следующей группе следует отнести вокодеры, основанные на модели речевого тракта, описываемой следующей сверткой:

$$y(t) = u(t) * s(t),$$

где

$y(t)$ - речевой сигнал;

$u(t)$ - сигнал возбуждения, возникающий в голосовых связках;

$s(t)$ - передаточная характеристика речевого тракта.

Исходя из данной модели, используя линейный предиктивный анализ производится вычисления авторегрессионных коэффициентов, которые описывают передаточную характеристику речевого тракта. Для передачи остаточного сигнала возбуждения применяются либо различные способы фильтрации и прореживания, уменьшающие объем необходимых для передачи данных, либо для остаточного сигнала возбуждения находится наиболее близкий аналог в некоторой наперед заданной кодовой книге и по каналу передачи отправляется только индекс в адресной книге. Данные вокодеры характеризуются значительной степенью сжатия речевого сигнала. К этому классу относятся следующие кодеки: G.723.1 (со скоростями передачи 5333 и 6400 Бит/сек.), G.728, G.729 и GSM 6.10, MELP, FS-1016.

Отдельное положение занимают схемы кодирования MPEG-1 и MPEG-2 Layer-3 (стандарты ISO 11172-3 и ISO 13818). Данные схемы разрабатывались для передачи музыкальной информации. Общая схема функционирования кодеков серии MPEG состоит в вычислении спектрального представления сигнала. Используя психоакустическую модель человеческого слуха, из сигнала в спектральной области удаляются элементы, которые в соответствии с моделью не воспринимаются человеческим ухом. Оставшаяся информация подвергается неравномерному кодирования коэффициентов которые в свою очередь могут сжиматься кодером Хаффмана для еще большего уменьшения потока данных.

Под растровым графическим изображением принято понимать двухмерную матрицу, элементами которой являются растры, характеризующие цвет точки изображения, принадлежащий множеству допустимых цветов (так называемому, цветовому пространству) [94]. Все растровые изображения принято делить по способу представления цветового пространства на следующие классы:

- мнохромные изображения: - это самый "простой" класс растровых изображений. В этих изображениях любая точка может принимать только два значения (это могут быть любые цвета, но наиболее распространены изображения из черных и белых точек). Для хранения информации об одной точке таких изображений достаточно 1 бита данных. Изображения данного класса сегодня не пользуются большой популярностью и используются лишь в некоторых системах распознавания образов.

- изображения с палитрой основаны на устаревшем способе представления цвета в растровых изображениях, но изображения данного класса весьма популярны и распространены в сети Интернет. Использование палитры (еще говорят – отображения цветов) в графических форматах связано с попыткой уменьшить размер хранимой информации. Вообще говоря, "палитра" впервые была применена в графических адаптерах, для упрощения их устройства и обеспечения большего разрешения при меньшем объеме оперативной памяти графического адаптера. Вслед за этим появились форматы хранения растровых графических изображений, основанные на использовании палитры, некоторые из которых активно используются и в наши дни. Ярким примером такого формата может служить GIF, который получил широкое распространение в сети Интернет и является неотъемлемой частью дизайна современных веб-страниц и Интернет рекламы. Количество передаваемых по сети файлов в формате GIF более чем в два раза превышает количество передаваемых страниц и писем (на смену устаревающего формата был разработан формат PNG, также позволяющий использовать палитру цветов, однако, он еще не получил большого распространения). Палитра цветов – это набор из n элементов, каждый из которых задает (как и точка обычного полноцветного изображения) интенсивность цветовых составляющих в каком-либо фиксированном цветовом пространстве (обычно RGB), при этом каждая точка изображения содержит лишь номер цвета из палитры, а не информацию о ее цвете в цветовом пространстве.

- полноцветные изображения – это самый распространенный класс изображений. Увеличивающаяся доступность для широкого круга пользователей устройств оцифровки изображений (сканеров, цифровых фотоаппаратов, web-камер), неизменно приводит к резкому увеличению всевозможных изображений данного класса, обладающих различными характеристиками и отличительными особенностями. Каждая точка растрового графического изображения данного класса задает интенсивность цветовых составляющих в каком-либо фиксированном цветовом пространстве (RGB, CMY, CMYK и т.д.).

По способу происхождения, как аудио-сигналы, так и графические изображения принято делить на оцифрованные и неоцифрованные (полученные при помощи компьютерных программ).

Под *пространством сокрытия* (*ПС*) будем понимать линеаризованную последовательность элементов мультимедиа-контейнера, пригодных для сокрытия, которые в соответствии с его типом и форматом кодируют мультимедиа-данные.

Поясним введенное понятие на примере конкретных типов и форматов мультимедиа контейнеров. Для аудио-сигналов, формат хранения которых не использует сжатие с потерями, пространством сокрытия является линеаризованная последовательность отсчетов. Для полноцветных графических изображений, формат хранения которых не использует сжатие с потерями, пространством сокрытия является линеаризованная последовательность элементов цветового пространства всех точек. Для полноцветных графических изображений сжатых с потерями информации при помощи стандарта JPEG пространством сокрытия является линеаризованная последовательность коэффициентов, полученных после этапа квантования.

Пусть b – это контейнер, а m – это скрываемое сообщение, и соответственно $|b|$ – это объем контейнера, а $|m|$ – размер сообщения.

Используемым пространством сокрытия (*ИПС*) будем называть совокупность элементов пространства сокрытия, в которые действительно произошло внедрение информации. Обозначив для удобства файл, содержащий контейнер через *File*, получим следующую цепочку соотношений:

$$|m| \leq |ИПС| \leq |ПС| \leq |b| \leq |File|.$$

Определим *коэффициент сокрытия* как отношение размера сообщения к размеру контейнера $KС = \frac{|m|}{|b|}$, а *плотность сокрытия* как отношение размера используемого пространства сокрытия ко всему пространству сокрытия $\rho = \frac{|ИПС|}{|ПС|}$.

2.1.2. Разработка базовой модели естественности мультимедиа-информации

Разработанная модель естественности состоит в том, что для каждого мультимедиа-контейнера вычисляется набор параметров [63]:

$$r = (r_0, r_1, \dots, r_n), \text{ где } n \text{ – количество параметров.}$$

Данный набор параметров при условии их попадания в определенные "естественные" интервалы характеризует естественность мультимедиа-контейнера. Для определения естественности делается ряд допущений. Предполагается, что существует достаточно большой набор эталонных мультимедиа-контейнеров, которые считаются естественными и задают множество естественности G . Для произвольного мультимедиа-контейнера I необходимо вычислить набор связанных с ним параметров r , и проверить принадлежность этого набора множеству естественности G .

Опишем основные шаги алгоритма проверки естественности мультимедиа-контейнера и определения максимально допустимой плотности сокрытия:

1. Для каждого эталонного мультимедиа-контейнера вычисляется набор связанных с ним параметров.
2. Для каждого мультимедиа-контейнера производится сокрытие в него информации со всеми допустимыми плотностями сокрытия.
3. Для каждого модифицированного мультимедиа-контейнера вычисляется набор связанных с ним параметров.
4. Полученные наборы параметров анализируются совместно с эталонным набором параметров и вычисляется значение максимально допустимой плотности сокрытия.

Действие большинства современных стeganографических систем не приводит к визуальному ухудшению контейнеров и в общем случае не может быть статистически выявлено на фоне естественных шумов присутствующих в оцифрованных изображениях. Однако, если разбить все мультимедиа-контейнеры на классы и для каждого из них построить модель естественности, то может оказаться, что сокрытие может быть обнаружено в предположении, что контейнер принадлежит к тому или иному классу.

Для оценки возможности автоматического принятия решения о наличии в мультимедиа-контейнере сокрытой информации используется решающее правило, основанное на критерии Неймана-Пирсона [50]. При составлении решающего правила, было сделано предположение о нормальном распределении значений характеристик:

$$r = (r_0, r_1, \dots, r_n), \text{ где } n - \text{ количество параметров},$$

для пустого и заполненного контейнера. Использование этого предположения и полученных значений различных характеристик, позволяет получить значения ошибок первого и второго рода для различных характеристик и различных степеней сокрытия.

2.1.3. Параметры контроля пар значений различных элементов пространства сокрытия

Задача данного параметра, состоит в том, чтобы контролировать естественность взаимосвязи различных бит в отсчетах пространства сокрытия [49]. Данный параметр основан на том факте, что при сокрытии информации модифицируются значения отсчетов и нарушается существующая связь между различными битами в рамках отсчета.

Принцип работы современных стеганографических систем, использующих в качестве контейнеров мультимедиа-информацию, основан на факте присутствия в них шумов (которые возникают в силу квантования, цифровой обработки и применения алгоритмов сжатия). Говоря иначе, в основу некоторых методов сокрытия положено предположение о том, что младшие биты носят случайный характер. Это предположение верно для подавляющего большинства оцифрованных мультимедиа-контейнеров в том случае, если рассматривать младшие биты отсчетов как отдельный сигнал, не зависящий от остальных бит отсчетов s_i (см. рисунок 5).

7	6	5	4	3	2	1	0
1	0	0	0	1	1	0	1

s_i

Рисунок 5 – Восьмибитный отсчет

В общем случае, шумы различны для различного уровня входного сигнала, и как правило, увеличиваются при его уменьшении. В качестве примера можно взять современные планшетные сканеры: чем темнее сканируемое изображение, тем сильнее видны шумы оцифровки.

Поэтому анализ шумов нужно проводить, учитывая значения всех бит в отсчете пространства сокрытия, а не только младших. Для этого необходимо разделить все биты отсчета s_i на две части, мы обозначим их s'_i и s''_i . В левую часть s'_i включены старшие биты отсчета, поэтому она характеризует уровень амплитуды сигнала. В правую часть s''_i –

младшие биты, в силу чего можно считать, что она характеризует шумы с амплитудой заданной s_i^l . Не нарушая общности, будем считать что отсчеты пространства скрытия являются 8-ми битными. При выборе только одного младшего бита в качестве правой части s_i^r , возможно проводить анализ шумов с учетом не более чем 128 различных уровней сигнала (см. рисунок 6). Однако, если шумы в контейнере велики, то более уместным является выбор в качестве правой части двух младших битов, при котором максимальное количество уровней сигнала уменьшается для 64 (см. рисунок 7).

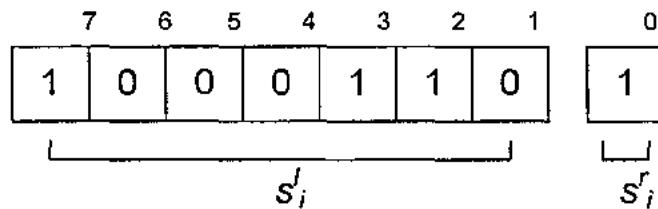


Рисунок 6 – Восьмибитный отсчет, с однобитной правой частью s_i^r

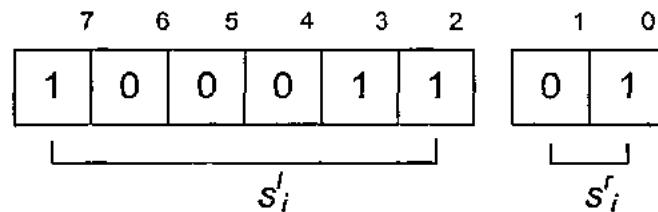


Рисунок 7 – Восьмибитный отсчет, с двухбитной правой частью s_i^r

Пусть $p_{s_i}(s^r)$ - это частота появления различных значений s^r при различных уровнях сигнала s^l . В нашем случае формула для ее вычисления имеет вид:

$$p_{s_i}(s^r) = p(s^l, s^r) = \frac{|\{i | s_i^l = s^l \& s_i^r = s^r\}|}{|\{i | s_i^l = s^l\}|}.$$

В качестве параметров естественности можно использовать весь набор частот появления различных значений (см. рисунки 8 и 9). Однако, если использование большого количества параметров нежелательно и известно, что скрываются зашифрованные сообщения (т.е. вероятность появления 1 на любом участке сообщения равна 1/2), то возможно уменьшение количества параметров при помощи применения теста Хи-квадрат:

$$p_{s_i}^* = p^*(s^l) = \frac{|\{i | s_i^l = s^l\}|}{|s^r|},$$

$$\chi^2(s') = \sum_s \frac{(p_{s'}(s') - p_s)^2}{p_s},$$

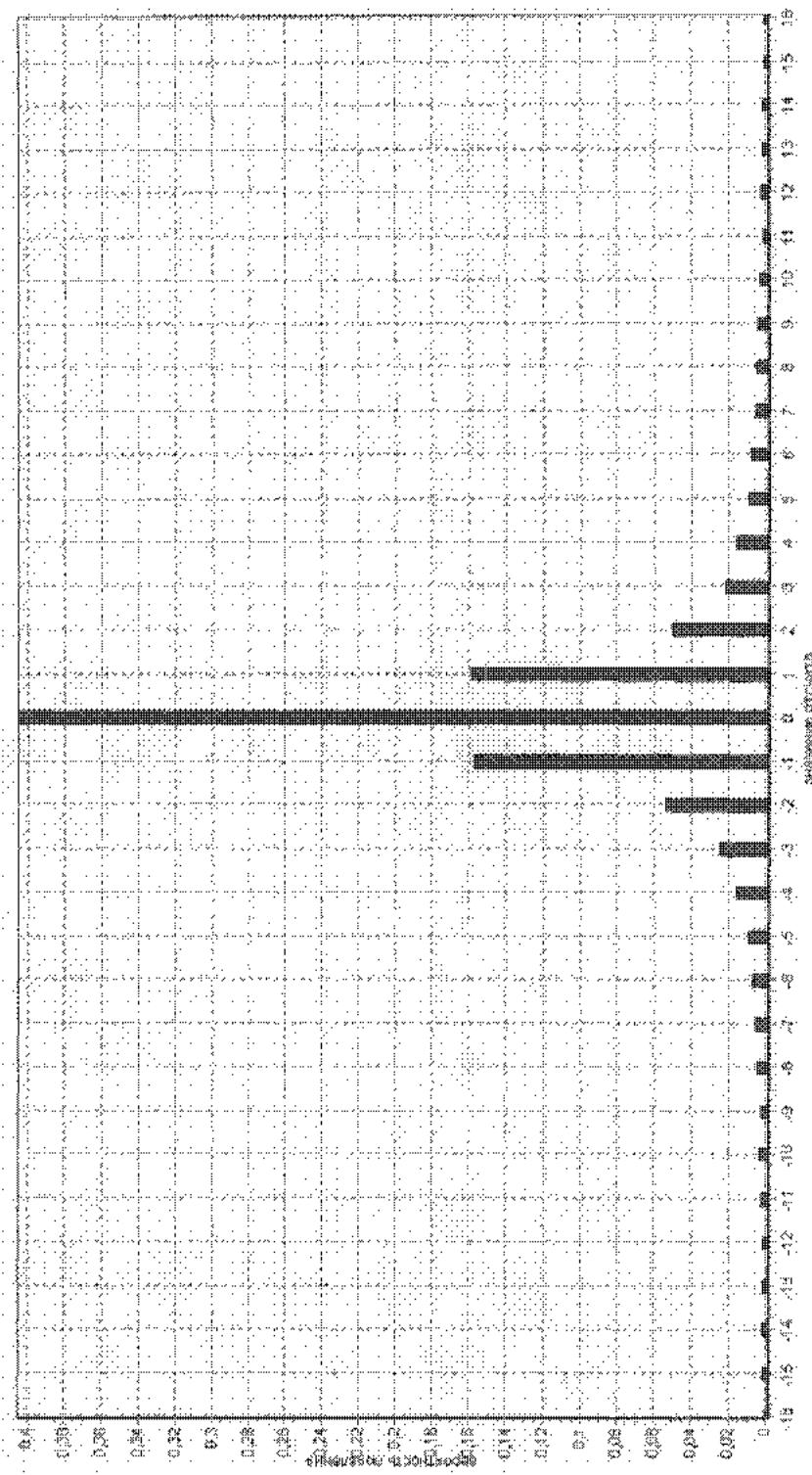


Рисунок 8 – Распределение частот встречаемости различных отсчетов немодифицированного контейнера

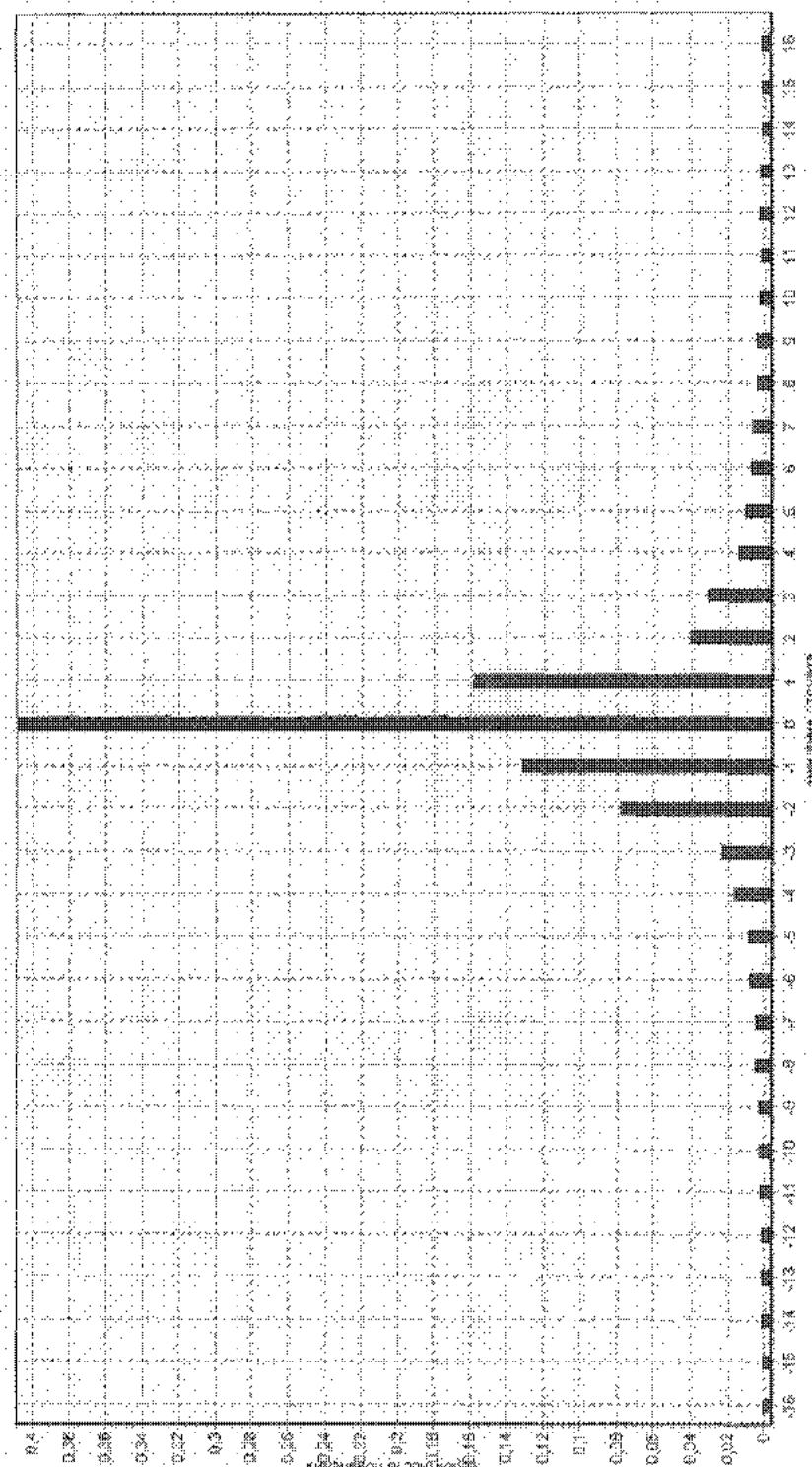


Рисунок 9 — Распределение частот встречаемости различных отсчетов модифицированного контейнера

2.1.4. Параметры контроля частот серий битовых значений различных отсчетов контейнера

Анализ частот серий битовых значений различных отсчетов мультимедиа-контейнера позволяет получить численные оценки статистических искажений, которые вносятся при сокрытии и нарушают зависимости между рядом стоящими отсчетами пространства сокрытия контейнера [81].

Пусть l - длина битовой серии, через n_i , $i=1..2^l$ обозначим вероятность появления серии i в пространстве сокрытия мультимедиа-контейнера.

Так как при сокрытии биты отсчетов пространства сокрытия заменяются на биты скрываемого сообщения, которые равновероятны для сокрытия шифрованной информации, то ожидаемая вероятность серии i определяется как:

$$n_i^1 = \frac{1}{2^l}.$$

Для описания естественности частот серий битовых значений различных отсчетов контейнера воспользуемся следующей статистикой χ^2 с k степенями свободы:

$$\chi^2_{k-1} = \sum_{i=1}^k \frac{(n_i - n_i^1)^2}{n_i^1}.$$

На рисунках 10 и 11 приведены примеры распределений частот серий, получаемых для пустых и модифицированных изображений в формате JPEG.

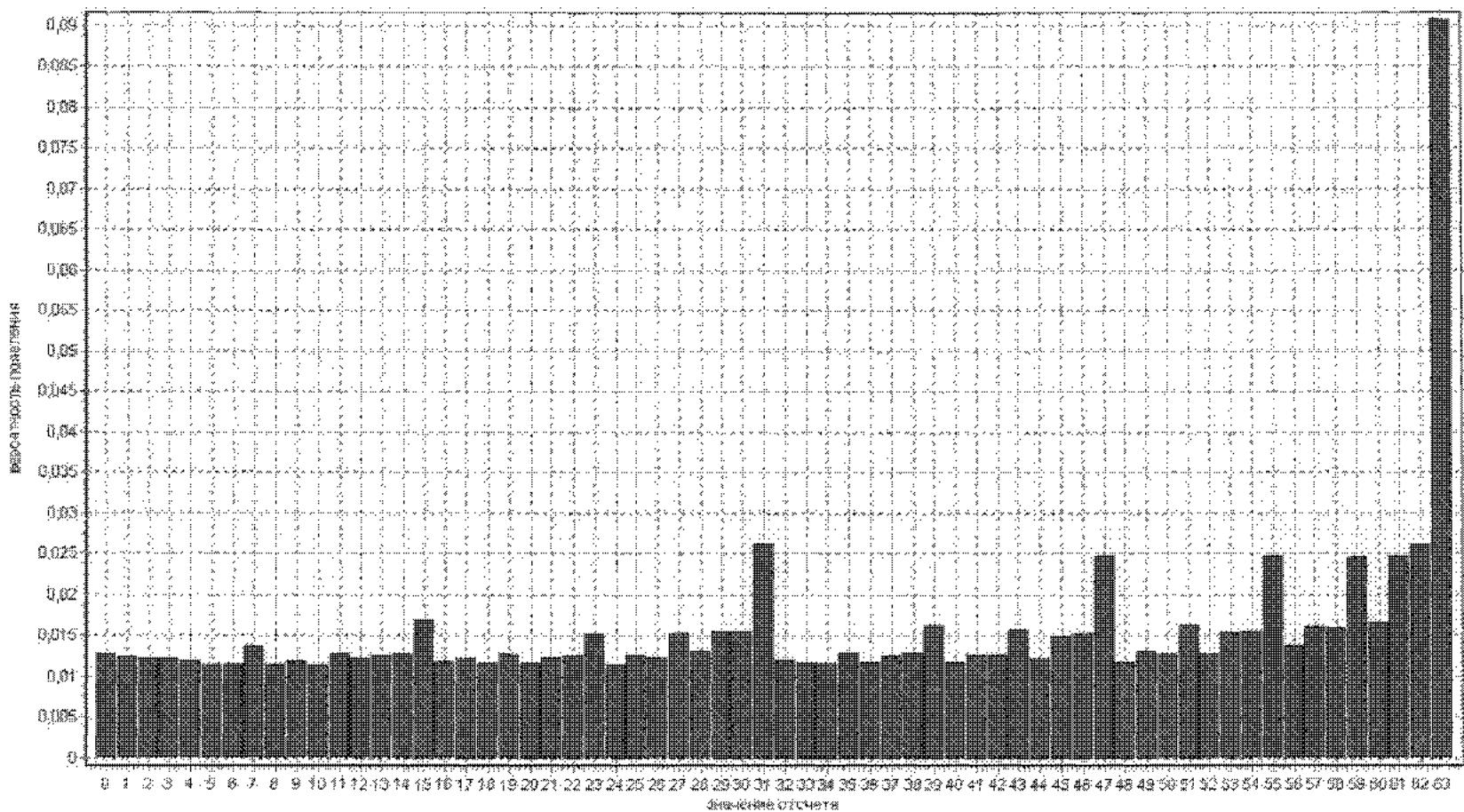


Рисунок 10 – Распределение частот встречаемости серий немодифицированного изображения при лине серии рангом б

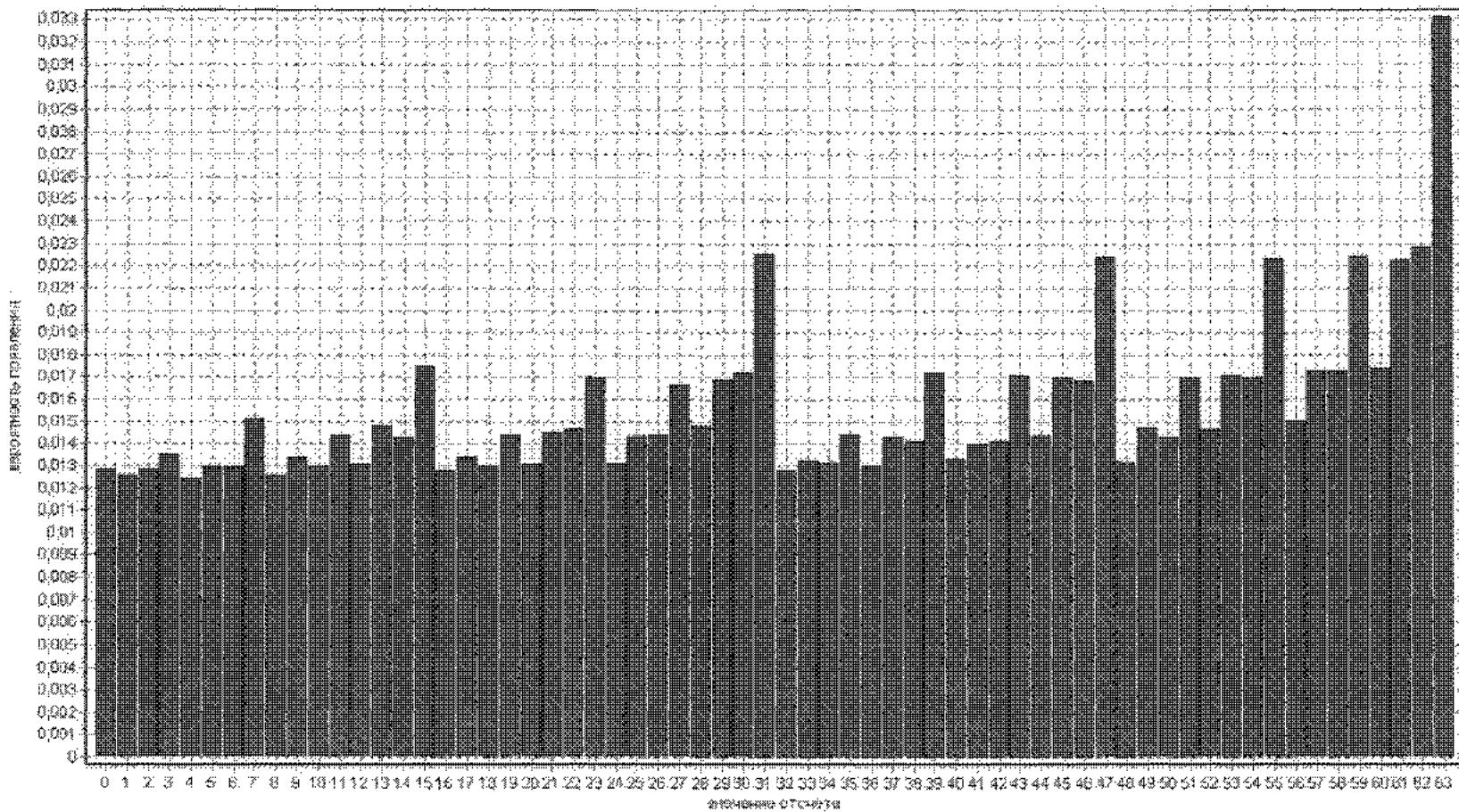


Рисунок 11 – Распределение частот встречаемости серий модифицированного изображения при длине серии равной 6

2.1.5. Параметры контроля длин битовых серий различных отсчетов контейнера

Анализ длин битовых значений различных отсчетов пространства сокрытия мультимедиа-контейнеров, как и анализ частот серий битовых значений различных отсчетов пространства сокрытия позволяет получить численные оценки статистических искажений, которые вносятся при сокрытии и нарушают зависимости между рядом (отметим, что для данного теста рассматриваются более отдаленные отсчеты, чем для предыдущего) стоящими отсчетами пространства сокрытия мультимедиа-контейнера.

Через n_i , $i=1..l$ обозначим вероятность появления серии из i одинаковых бит в пространстве сокрытия мультимедиа-контейнера.

Ожидаемая вероятность серии из i одинаковых бит в случае равновероятных значений определяется как:

$$n_i^1 = \frac{1}{2^i}, i=1..l.$$

Для описания естественности длин битовых значений различных отсчетов контейнера воспользуемся следующей статистикой χ^2 с k степенями свободы:

$$\chi^2_{k-1} = \sum_{i=1}^k \frac{(n_i - n_i^1)^2}{n_i^1}.$$

На рисунках 12 и 13 приведены примеры распределений длин битовых серий, получаемых для пустых и модифицированных изображений в формате JPEG.

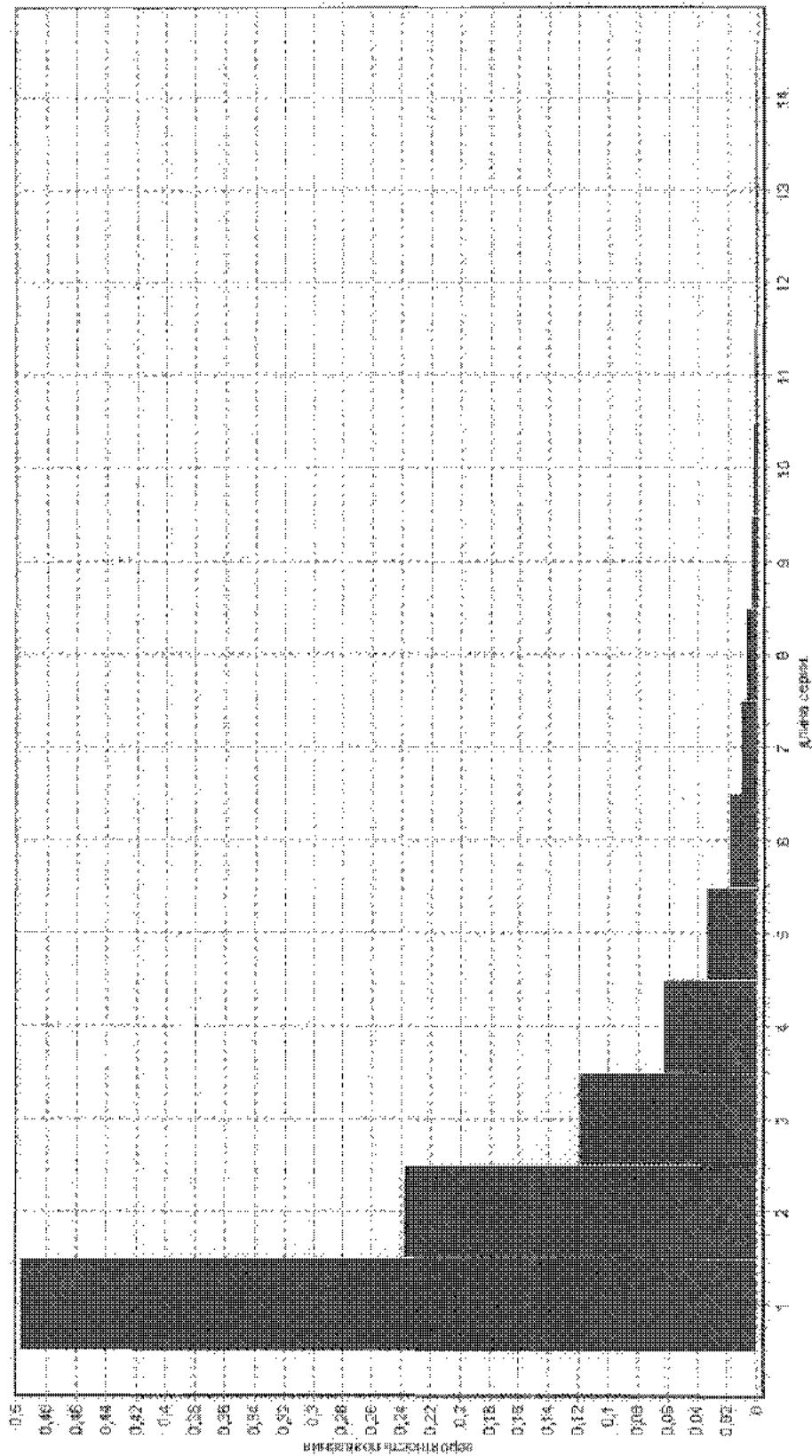


Рисунок 12 – Распределение длин битовых серий немодифицированного изображения в формате JPEG

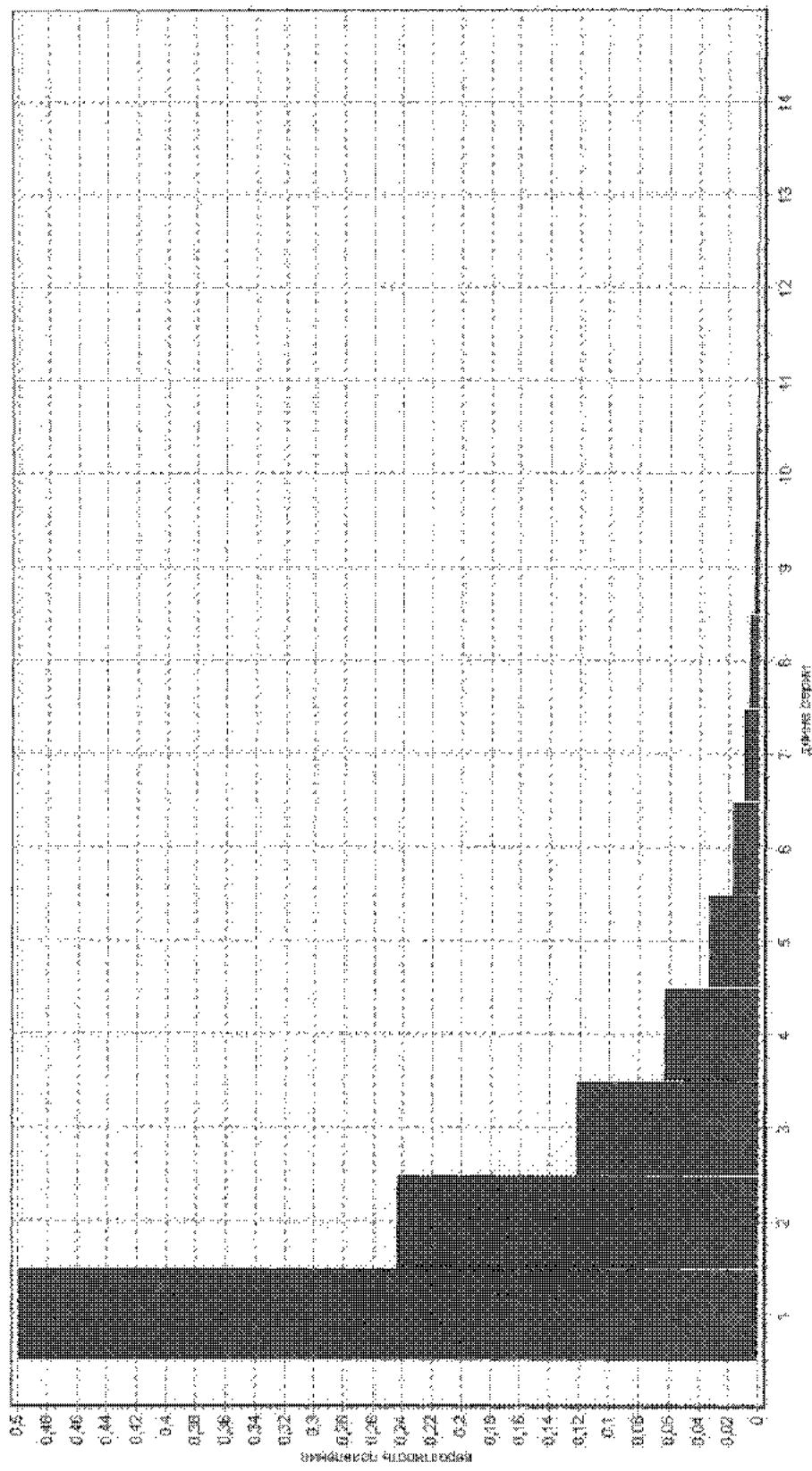


Рисунок 13 – Распределение длин битовых серий модифицированного изображения в формате JPEG

2.2. Разработка методов внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров

Процесс внедрения цифровых водяных знаков в мультимедиа-контейнеры согласно разработанному методу разделен на следующие этапы [45]:

- определение типа контейнера;
- определение пространства сокрытия;
- подготовка цифрового водяного знака к сокрытию;
- сокрытие данных.

Очередность выполнения этапов сокрытия цифрового водяного знака показана на рисунке 14. Рассмотрим каждый этап в отдельности.



Рисунок 14 – Общая схема выполнения этапов внедрения ЦВЗ

Определение типа контейнера

Целью этапа определения типа контейнера является определение формата хранения мультимедиа-контейнера, который предполагается использовать для внедрения ЦВЗ. Данный этап одинаков как для сокрытия, так и для извлечения и заключается в выполнении следующих шагов:

- определение типа мультимедиа-контейнера (звук, изображения, видео);
- определение формата хранения мультимедиа-контейнера;
- определение свойств мультимедиа-контейнера (количество каналов, разрядность отсчетов, высоты, ширины, цветового пространства, наличие палитры, и т.п. в зависимости от типа и формата).

На данном этапе осуществляется предварительный контроль пригодности контейнера для сокрытия информации, путем проверки соответствия его содержимого типу и формату хранения. Параметры, определенные на этом этапе используются для проверки контейнера на пригодность и определения пространства сокрытия.

Определение пространства сокрытия

Любое сокрытие двоичных данных в теле цифрового контейнера приводит к изменению некоторого числа бит контейнера. Выбор преобразования, используемого для сокрытия, влияет только на количество и распределение изменившихся бит в контейнере, и, как следствие, на статистические свойства контейнера и параметры, характеризующие его естественность . Для того чтобы построить метод внедрения информации, минимально изменяющий естественность контейнера, необходимо минимизировать число измененных в нем бит. Так как в этом случае искажения, вносимые в контейнер системой внедрения ЦВЗ, будут минимальны, а значит, будет минимально изменение статистик, характеризующих естественность изображения. Очевидно, что пространство сокрытия должно состоять только из тех бит, изменение которых приводит к минимальным визуальным искажениям в изображении (численно это искажения характеризуют метрики контроля качества, вычисленные между пустым и модифицированным контейнером) .

Рассмотрим сначала изображения, формат которых не содержит сжатие с потерями информации и палитру цветов. Очевидно, что наименьшим искажением в этом случае будет изменение одного младшего бита цветовой интенсивности отсчета. Изменение следующего за ним бита отсчета, является эквивалентным с точки зрения двоичных

сигналов, однако вызывает большие искажения, с учетом перцептивного восприятия изображения человеком, так как этот бит кодирует большую амплитуду цветовой интенсивности точки изображения. Рассуждая таким же образом, получим, что преобразованием, вызывающим наименьшие искажения в рассматриваемом классе изображений, является изменение одного из младших бит цветовых составляющих его отсчетов. Это преобразование является наиболее точным инструментом при "работе" с контейнером и следовательно, его выбор в качестве преобразования, используемого для сокрытия является оптимальным. Отметим, что пространство сокрытия, отвечающее данному преобразованию, состоит из младших битов всех цветовых составляющих всех точек изображения.

Рассуждая аналогичным образом, получим, что оптимальным пространством сокрытия для растровых графических изображений в формате JPEG являются младшие биты всех ненулевых отсчетов спектра изображения после квантования [107,93]. Это обусловлено тем, что изменение нулевых отсчетов приводит к ухудшению сжатия (появление единичных элементов в высокочастотной части спектра может быть использовано для определения факта наличия сокрытого сообщения). С точки зрения перцептивного восприятия изображения, изменения младших бит элементов спектра различной частоты приводят к практически одинаковым искажениям. Это является следствием квантования, построенного на основе особенностей восприятия изображения глазом человека.

В случае, когда изображение содержит палитру цветов, выбор младших бит в качестве пространства сокрытия невозможен. Упорядочим элементы палитры по возрастанию тройки (R, G, B). Для этого с каждым элементом палитры свяжем его вес равный $(65536*G + 256*R + B)$ и упорядочим элементы палитры по возрастанию этого веса. Отметим, что выражение для веса выбрано не случайно, а построено в соответствии с особенностями восприятия цветовых составляющих глазом человека. Затем просмотрим упорядоченную палитру и удалим пары $(2i, 2i+1)$, если модуль хотя бы одной разности между соответствующими цветовыми составляющими превышает заданный порог d . Оптимальным значением порога является $d=3$, так как при использовании больших значений сокрытие может привести к появлению заметных искажений, а использование меньших значений не желательно в силу снижения коэффициента эффективности стеганографической системы. Ясно, что в данном случае для сокрытия бита данных

необходимо изменить не младший бит цвета точки изображения, а все ее значение на новое, которое получается путем изменения младшего бита номера отсортированной палитры, в том случае, если он в ней присутствует. Говоря иначе, с каждым элементом исходной палитры, пригодным для сокрытия, необходимо связать два числа i и j_i , где i – исходный номер элемента палитры, j_i – номер i -го элемента палитры, полученный в результате ее сортировки и удаления непригодных пар. А сокрытие состоит в том, что по значению k точки изображения, пригодной для сокрытия, определяется соответствующий номер j_k . И его младший бит заменяется на скрываемый бит. Затем по получившемуся номеру j_k' определяется связанный с ним номер в исходной палитре номер k' , который и присваивается текущей точке. Таким образом, для изображений, формат которых использует палитру цветов, пространство сокрытия состоит из младших бит индексов отсортированной палитры точек изображения, пригодных для сокрытия.

Подготовка ЦВЗ к сокрытию

Скрываемый ЦВЗ представляет собой двоичный вектор фиксированной длины и имеет следующий формат (см. рисунок 15):

- код контроля целостности – значение хэш-функции, электронная цифровая подпись и.т.п.;
- служебные данные – информация характеризующая мультимедиа-контейнер.



Рисунок 15 – Общий формат ЦВЗ

Подготовка ЦВЗ к сокрытию состоит в вычислении значений всех полей, формированию двоичного вектора согласно приведенному формату и в последующем шифровании служебных данных с использованием ключа шифрования.

Для того, чтобы ЦВЗ могло быть использовано для контроля целостности мультимедиа информации, идентификации контейнеров и их копий в хранилищах данных необходимо использовать двоичный вектор фиксированной длины (384 бита) следующего формата (см. рисунок 16):

- код контроля целостности (256 бит) – значение хэш-функции;

- идентификатор создателя (32 бита, уникальный номер);
- идентификатор контейнера (32 бита, уникальный номер);
- идентификатор копии контейнера (32 бита, уникальный номер);
- идентификатор категории контейнера (32 бита, уникальный номер).

код контроля целостности	идентификатор создателя	идентификатор контейнера	идентификатор копии	идентификатор категории
256 бит	32 бита	32 бита	32 бита	32 бита

Рисунок 16 – Пример реализации ЦВЗ

Сокрытие и извлечение данных

Опишем разработанный метод сокрытия данных в растровых графических файлах и алгоритм его выполнения. Будем считать, что задан произвольный стеганографический ключ k , сообщение m , и мультимедиа-контейнер B , пригодный для сокрытия. Через m' обозначим данные, полученные после подготовки сообщения m к сокрытию. В зависимости от типа и формата мультимедиа-контейнера определим пространство сокрытия, задав упорядоченную последовательность его элементов b_i , $i=1..n$. Согласно разработанному методу, на основе стеганографического ключа k при помощи генератора псевдо-случайных чисел (ГПСЧ) создается стеганографическая ключевая последовательность k_i , которая при заданной плотности контейнера однозначно определяет способ сокрытия бит сообщения в контейнере [44].

Через $p \in N$ обозначим плотность контейнера, определяющую количество бит ключевой последовательности, которое ставится в соответствие одному элементу пространства сокрытия. Физический смысл введенной величины состоит в управлении плотностью сокрытия: чем больше плотность контейнера, тем меньше информации в нем можно скрыть, т.е. тем меньше плотность сокрытия.

Так как размер скрываемого сообщения фиксирован и равен $|m|$ бит, то плотность сокрытия определяется по формуле:

$$p = \lceil \log_2 \frac{9n}{10|m|} \rceil.$$

В соответствии с произвольной фиксированной плотностью контейнера p разобьем ключевую последовательность на блоки $k_{p,l}$ с размером p бит каждый, при этом между

полученными блоками $k_{p,i}$ и упорядоченными элементами пространства сокрытия b_i устанавливается взаимно однозначное соответствие.

После чего выбирается шаблон sh , представляющий собой двоичный вектор с размером p бит. Сокрытие производится только в те элементы b_i пространства сокрытия, которым соответствуют блоки ключевой последовательности, совпадающие с выбранным шаблоном, т.е. для которых выполнено $k_{p,i}=sh$.

Перед тем, как определить способы выбора шаблона sh , укажем требования, которым они должны удовлетворять:

- выбор шаблона не должен зависеть от скрываемых данных и данных контейнера;
- выбор шаблона должен зависеть от стеганографического ключа;
- выбор шаблона должен быть одинаков для этапов сокрытия и извлечения данных;
- частота появления шаблона среди блоков ключевой последовательности не должна отличаться от теоретической более чем на 5%, т.е. должно быть выполнено:

$$\frac{0.95}{2^p} \leq \frac{|\{k_{p,j} : k_{p,j} = sh, 1 \leq j \leq n\}|}{n} \leq \frac{1.05}{2^p} \quad (2.1)$$

Опишем два способа выбора шаблона. Первый способ выбора шаблона состоит в следующем: в качестве шаблона берется $n+1$ блок ключевой последовательности: $sh=k_{p,n+1}$, после чего выполняется проверка неравенства (2.2.1). Если неравенство не выполняется, то в качестве шаблона берется очередной блок ключевой последовательности и выполняется его проверка на пригодность. Так повторяется до тех пор, пока не будет найден подходящий шаблон. Достоинством данного алгоритма является простота его реализации, а недостатком необходимость в доказательстве его конечности.

Второй способ состоит в том, что для всех блоков ключевой последовательности вычисляется отклонение частоты их появления от теоретической:

$$\Delta k_{p,i} = \left| \frac{|\{k_{p,j} : k_{p,j} = k_{p,i}, 1 \leq j \leq n\}|}{n} - \frac{1}{2^p} \right|, \quad i = 1, 2, \dots, n.$$

Шаблон sh выбирается среди блоков $k_{p,q}$ с наименьшим отклонением. В качестве шаблона выбирается тот блок с наименьшим отклонением, который встретился в ключевой последовательности раньше всех остальных блоков с наименьшим отклонением. Затем выполняется проверка неравенства (2.2.1), оно должно быть выполнено в силу свойства монотонности ГПСЧ.

При выборе шаблона sh описанными выше способами, плотность сокрытия приближению будет равна $1/2^p$ (отметим, что это выражение и определяет ее связь с плотностью контейнера).

Прямое стеганографическое преобразование $F:M \times B \times K \rightarrow B$ для разработанного метода сокрытия данных в мультимедиа-информации имеет вид:

$$b'_i = \begin{cases} b_i, & \text{если } k_{p,i} \neq sh \\ m_i, & \text{если } k_{p,i} = sh, \text{ где } l = |\{k_{p,j} : k_{p,j} = sh, j \leq i\}| \end{cases} \quad i=1, 2, 3, \dots, n$$

Извлечение ЦВЗ происходит в соответствии с сокрытием и состоит из следующих этапов:

- определение типа контейнера;
- определение пространства сокрытия;
- извлечение ЦВЗ.

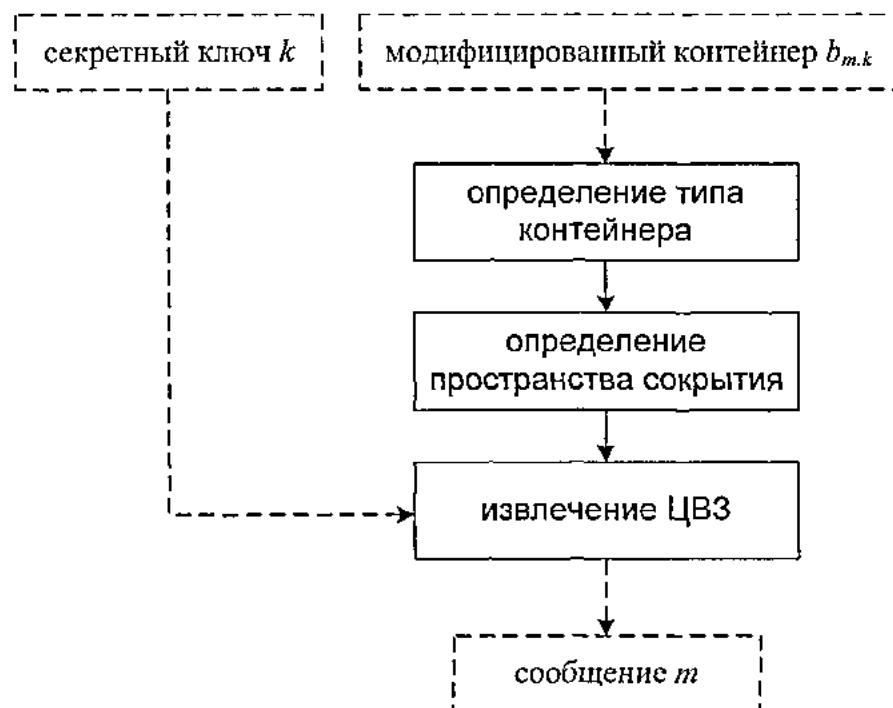


Рисунок 17 – Общая схема выполнения этапов извлечения ЦВЗ из мультимедиа-контейнера

Очередность выполнения этапов извлечения сообщения показана на рисунке 17.

Извлечение ЦВЗ состоит в применении **обратного стеганографического преобразования** $F^{-1}:B \times K \rightarrow M$ которое построено в соответствии с разработанным методом сокрытия:

$$m'_i = b'_l, \text{ где } l : |\{k_{p,i} : k_{p,l} = sh, j \leq l\}| = i, \quad i = 1, 2, 3, \dots, n.$$

2.3. Разработка метода контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков

Метод контроля целостности мультимедиа-контейнеров базируется на использовании стандартных хэш-функций. Как было сказано выше, для разрабатываемых криптографических систем можно рекомендовать только односторонние хэш-функции с длиной хэш-значения не менее 256 бит. В частности, этим требованиям удовлетворяет алгоритм ГОСТ 34.11-94. Метод контроля целостности позволяет использовать в том числе и хэши-функции с секретным ключом, для практического использования которых необходимо, чтобы длина ключа и хэш-значения составляла не менее 128 бит, однако, в настоящее время возможно использование функций с длиной ключа не менее 128 бит и длиной хэш-значения не менее 64-х бит (например, ГОСТ 28147-89 (режим 4)).

Таким образом, возможно использование хэши-функции произвольного типа, при условии внесение соответствующих изменений в формат ЦВЗ.

Перейдем к формальному описанию разработанного метода.

Разработанный метод контроля целостности мультимедиа-информации на основе извлеченных ЦВЗ состоит из двух составляющих:

- внедрения ЦВЗ с кодом контроля целостности;
- проверки целостности мультимедиа-информации.

Алгоритм внедрения ЦВЗ с кодом контроля целостности состоит в выполнении следующих шагов:

1. Пусть I – это исходный мультимедиа-контейнер, k – стеганографический ключ. Через $h(k)$ обозначим хэш-значение вычисление от стеганографического ключа. Пусть m – это ЦВЗ, подготовленный к сокрытию, содержащий в качестве хэш-значения $h(k)$. Внедрим в контейнер ЦВЗ $I_{m,k}=F(m, I, k)$.

2. Вычислим код контроля целостности контейнера $h(I_{m,k})$, таким образом что хэш-значение вычисляется непосредственно от данных кодирующих мультимедиа-информации (которые включают пространство сокрытия).

3. Заменим в ЦВЗ m хэш-значение на $h(I_{m,k})$ и повторно внедрим в контейнер ЦВЗ, содержащий его код контроля целостности $I_{m,k}=F(m, I, k)$.

Алгоритм проверки целостности мультимедиа-информации состоит в выполнении следующих шагов:

1. Извлечем из мультимедиа-контейнера ЦВЗ $m=F^{-1}(I_{m,k}, k)$.
2. Вычислим $h(k)$. Подготовим ЦВЗ m' , у которого поле кода контроля целостности содержит $h(k)$, а поле служебных данных совпадает с m .
3. Внедрим в контейнер ЦВЗ m' : $I_{m',k}=F(m', I_{m,k}, k)$.
4. Вычислим код контроля целостности контейнера $h(I_{m',k})$.
5. Сравним вычисленное значение кода контроля целостности $h(I_{m',k})$ с извлеченным $h(I_{m,k})$.

2.4. Разработка модели оценки вносимых при внедрении искажений и методов их количественной оценки

Разработку модели оценки вносимых искажений начнем с формализации определения *критерия качества* мультимедиа-информации. Под критерием оценки качества мультимедиа-контейнера будем понимать метрику между исходным (неискаженным, обозначим его I) и модифицированным (искаженным, обозначим его O) мультимедиа-контейнером [48]:

$$q = \mu(I, O).$$

С точки зрения контроля искажений, вносимых стеганографическими системами в графические изображения, исмодифицированный и модифицированный контейнер представляют собой неискаженное и искаженное изображение соответственно.

Для оценки искажений, вносимых при внедрении ЦВЗ, может быть достаточно всего лишь одной метрики, которая в конкретных условиях эксплуатации адекватно характеризует перцептивное восприятие ухудшения качества. Однако при оценке искажений для широкого спектра всевозможных типов и форматов хранения мультимедиа-информации нарушается адекватность модели.

Разработанная модель оценки представляет собой набор критериев оценки качества мультимедиа-контейнеров: $S = \{\mu_i(I, O), i=1\dots n\}$, где n – размерность модели. Для оценки искажений в соответствии с типом и форматом мультимедиа-контейнера выбирается максимальное возможное подмножество имеющихся критериев оценки качества $S = \{\mu_i(I, O) | I, O \in B\}$, где B – заданное множество мультимедиа-контейнеров.

При рассмотрении метрик, формирующих модель, не нарушая общности, ограничимся рассмотрением растровых графических изображений. Условимся, что растровые графические изображения - это матрицы, элементами которых являются точки (растры) изображения. Каждая точка принадлежит цветовому пространству, в котором представлено изображение. Сегодня используется целый ряд различных цветовых пространств, самыми распространенными из которых являются RGB, YCrCb, CMY, CMYK. От правильного выбора цветового пространства зависит успех в применении метрик искажений на практике. Для упрощения формул в дальнейшем будем рассматривать одномерное цветовое пространство (например, как в монохромных изображениях). Через $I_{i,j}$ и $O_{i,j}$ будем обозначать значение точки с координатами (i,j) неискаженного и искаженного изображения соответственно.

Отметим, что для формирования модели используются распространенные метрики [metric1, metric2], адаптированные для различных типов мультимедиа-информации. Подобный подход, в отличии от традиционных, приводит не к одному значению, а к вектору значений, характеризующему происходящие изменения.

Самый простой способ осуществлять контроль искажений заключается в вычислении наибольшего и наименьшего отклонений:

$$\mu_{MAX}(I, O) = \max_{i,j} |I_{i,j} - O_{i,j}|, \quad \mu_{MIN}(I, O) = \min_{i,j} |I_{i,j} - O_{i,j}|.$$

Еще одна метрика контроля искажений получается вычислением среднего по модулю отклонения:

$$\mu_{AD}(I, O) = \frac{1}{MN} \sum_{i,j} |I_{i,j} - O_{i,j}|.$$

Нормированное среднее по модулю отклонение есть отношение среднего по модулю отклонения к среднему по модулю исходного изображения:

$$\mu_{NAD}(I, O) = \frac{\sum_{i,j} |I_{i,j} - O_{i,j}|}{\sum_{i,j} |I_{i,j}|}.$$

Наибольшее, наименьшее, среднее по модулю и нормированное среднее по модулю отклонения являются наиболее грубыми оценками искажений, вносимых в графические изображения. Однако они весьма просты в вычислении и могут быть полезны на практике. Например, если разбить изображение на блоки, и вычислить эти метрики отдельно для каждого блока, то можно проводить анализ распределения искажений по плоскости изображения с минимальными вычислительными затратами. Для осуществления более точного контроля возможно использование квадратичных отклонений.

Средне-квадратичное отклонение:

$$\mu_{MSE}(I, O) = \frac{1}{MN} \sum_{i,j} (I_{i,j} - O_{i,j})^2.$$

Нормированное среднее квадратичное отклонение:

$$\mu_{NMSE}(I, O) = \frac{\sum_{i,j} (I_{i,j} - O_{i,j})^2}{\sum_{i,j} (I_{i,j})^2}.$$

Основное отличие квадратичных отклонений от линейных состоит в том, что большие по модулю разности интенсивностей вносят более существенный вклад в результирующую метрику - в теории статистики такие разности называют "выбросами". На самом деле, чем больше порядок статистического момента и выше степень - тем сильнее влияют на результат не вписывающиеся в общее распределение выбросы. Квадратичные параметры контроля лучше отражают восприятие изображения человеком: большие по модулю точечные искажения могут быть замечены при просмотре с большей вероятностью и поэтому малое количество больших отклонений менее предпочтительно, чем большее количество малых.

На практике, часто удобней использовать две другие "формы" нормированного среднего квадратичного отклонения. Отношение сигнал-шум, в котором в качестве сигнала берется неискаженное изображение, а шум определяется как поточечная разность между неискаженным и искаженным изображением соответственно:

$$\mu_{SNR}(I, O) = \frac{1}{\mu_{NMSE}(I, O)} = \frac{\sum_{i,j} (I_{i,j})^2}{\sum_{i,j} (I_{i,j} - O_{i,j})^2}.$$

Критерием точности изображения принято называть метрику, вычисляемую по формуле:

$$\mu_{HF}(I, O) = 1 - \mu_{NMSE}(I, O) = 1 - \frac{\sum_{i,j} (I_{i,j} - O_{i,j})^2}{\sum_{i,j} (I_{i,j})^2}.$$

А L_p -норма, пригодная для вычисления параметров контроля выглядит следующим образом:

$$\mu_{L_p}(I, O) = \left(\frac{1}{MN} \sum_{i,j} (I_{i,j} - O_{i,j})^p \right)^{1/p}.$$

Дело в том, что перцептивное восприятие искажений изображения не носит квадратичного характера, и сильно зависит от средней яркости окрестностей изменявшихся точек. L_p -норма при $p > 10$ мало полезна на практике, оптимальным значением параметра считается $p=4$. Полезно бывает разбивать изображения на блоки одинакового размера, и считать общую L_p -норму как сумму L_p -норм блоков, при этом для однотонных блоков лучше выбирать меньшие значения параметра p :

$$\mu_{L_p}(I, O) = \sum_b \left(\frac{1}{mn} \sum_{(i,j) \in b} (I_{i,j} - O_{i,j})^{p_b} \right)^{1/p_b}.$$

Средне-квадратичное отклонение лапласиана характеризует искажение не самого изображения, а его производной, что позволяет оценить изменение цветовых переходов (например, появление контрастных участков):

$$\mu_{LMSE}(I, O) = \frac{\sum_{i,j} (\nabla^2 I_{i,j} - \nabla^2 O_{i,j})^2}{\sum_{i,j} (\nabla^2 I_{i,j})^2},$$

$$\nabla^2 I_{i,j} = I_{i+1,j} + I_{i,j+1} + I_{i-1,j} + I_{i,j-1} - 4I_{i,j}$$

Пиковое отношение сигнал-шум, как и отношение сигнал-шум имеет большое практическое значение в современной обработке сигналов, многие разработчики ограничиваются лишь применением этих, известных всем, метрик:

$$\mu_{PSNR}(I, O) = MN \frac{\max_{i,j} (I_{i,j})^2}{\sum_{i,j} (I_{i,j} - O_{i,j})^2}.$$

Нормированная взаимная корреляция и корреляция качества – это самые распространенные статистические метрики, на практике первая из них может быть более полезной за счет сохранения размерности:

$$\mu_{NC}(I, O) = \frac{\sum_{i,j} I_{i,j} O_{i,j}}{\sum_{i,j} (I_{i,j})^2}, \quad \mu_{CQ}(I, O) = \frac{\sum_{i,j} I_{i,j} O_{i,j}}{\sum_{i,j} I_{i,j}}.$$

Конструктивный контент – квадрат отношения квадратичных норм изображений - еще одна из используемых в стеганографии метрик:

$$\mu_{SC}(I, O) = \frac{\sum_{i,j} (I_{i,j})^2}{\sum_{i,j} (O_{i,j})^2}.$$

Следующие три метрики применимы к изображениям, разбитым на блоки.
Глобальное отношение сигма-сигнала к шуму:

$$\mu_{GSSNR}(I, O) = \frac{\sum_b (\sigma_{I,b})^2}{\sum_b (\sigma_{I,b} - \sigma_{O,b})^2}.$$

Для блока b изображения I сигма-сигналом (сигма-сигнал от разности изображений $I-O$) можно рассматривать как метрику искажений) принято называть:

$$\sigma_{I,b} = \sqrt{\frac{1}{mn} \sum_{(i,j) \in b} (I_{i,j})^2 - \left(\frac{1}{mn} \sum_{(i,j) \in b} I_{i,j}\right)^2}.$$

$$\mu_{SSNR}(I, O) = \frac{1}{mn} \sum_b \mu_{SSNR,b}(I, O),$$

Отношение сигма-сигнала к шуму:

$$\mu_{SSNR,b}(I, O) = 10 \log_{10} \frac{\sigma_{I,b}^2}{(\sigma_{I,b} - \sigma_{O,b})^2}.$$

Отношение сигма-сигнала к средне квадратичному отклонению:

$$\mu_{SER,b}(I, O) = \frac{1}{nm} \frac{\sum_{(i,j) \in b} (I_{i,j})^2 - \frac{1}{mn} \left(\sum_{(i,j) \in b} I_{i,j}\right)^2}{\sum_{(i,j) \in b} (I_{i,j} - O_{i,j})^2}.$$

Подобие гистограмм – это метрика, определяемая равенством:

$$\mu_{HS}(I, O) = \sum_{c=0}^C |v_I(c) - v_O(c)|,$$

$v(c)$ – относительная частота

Она бывает весьма полезна, в случае, если стеганографическая система влияет на относительные частоты точек изображения (например, как в методах сокрытия в изображениях с палитрой).

2.5. Выводы

Проведенный в первой главе анализ современных систем внедрения цифровых водяных знаков, систем защиты информации и методов контроля целостности позволил выявить недостатки, которые сужают область их практического применения. Среди них слишком большие вносимые искажения, возможность определения факта наличия сокрытых в контейнере данных, возможность извлечения сокрытых данных при условии известного модифицированного контейнера и неизвестного секретного ключа, неустойчивость к фальсификации сокрытых данных.

Методы и модели, разработанные в данной главе, основываются на использовании существующих результатов и направлены на преодоление выявленных недостатков.

1. В данной главе разработана модель естественности мультимедиа-информации, предназначенная для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков. Разработанная модель может быть использована для различных типов и форматов хранения мультимедиа-информации за счет введенного в ней понятия пространства сокрытия. Данная модель позволяет не только проводить оценку стойкости к обнаружению сокрытой информации, ее вторым назначением является контроль естественности контейнеров перед внедрением в них ЦВЗ (определение пригодности контейнеров).

2. На основе модели естественности разработаны стойкие методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров. Данные методы зависят от секретного ключа, однозначно определяющего способ внедрения и извлечения информации и позволяют равномерно распределять внедряемую информацию по пространству сокрытия.

3. Разработан метод контроля целостности мультимедиа-контейнеров, основанный на внедрении специальных цифровых водяных знаков. Данный метод позволяет использовать стандартные хэш-функции, но при этом для осуществления контроля целостности не требуется хранить хэш-значение.

4. Разработана модель оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки. Данная модель позволяет получать разносторонние численные характеристики, предназначенные для проведения оценки ухудшения перцептивного качества контейнеров.

Разработанные в данной главе модели и методы позволяют перейти к решению следующей частной задачи диссертационного исследования – разработке обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации, решению которой посвящена следующая глава.

Глава 3. Архитектура программного комплекса контроля целостности мультимедиа-информации

В предыдущей главе был разработан математический аппарат и методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров, а также метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков.

В данной главе разрабатывается архитектура программного комплекса контроля целостности мультимедиа-информации.

Первый параграф данной главы посвящен описанию разработанной обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации, основанной на использовании программных агентов, реализующих разработанные методы внедрения ЦВЗ и контроля целостности мультимедиа-информации.

Во втором параграфе главы приведено описание системы универсальных интерфейсов взаимодействия управляющих и функциональных компонент, разработанных в соответствии с разработанной в первой части обобщенной архитектурой программного комплекса контроля целостности мультимедиа-информации и позволяющих производить подключение общих функциональных модулей к различным типам программных агентов.

В третьем параграфе проводится разработка открытых интерфейсов для подключения программного обеспечения стороннего разработчика, позволяющих производить изменение и наращивание функциональности комплекса в целом и отдельных его программных агентов за счет подключения модулей сторонних разработчиков.

В соответствии с поставленной целью диссертационного исследования, для решения задачи организации комплекса, необходимо разработать обобщенную архитектуру программного комплекса контроля целостности мультимедиа-информации, включающую интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения программного обеспечения стороннего разработчика.

Данная архитектура должна позволять осуществлять масштабирование комплекса и его настройку в соответствии с уже имеющейся информационно-вычислительной инфраструктурой [70]. Открытые интерфейсы должны позволять производить изменения и наращивание функциональности за счет подключения модулей сторонних разработчиков. Решению этой задачи и посвящена данная глава.

В соответствии с выявленными недостатками существующих систем защиты, архитектура разрабатываемой системы имеет распределенный характер. В ходе работы могут одновременно использоваться несколько компьютеров, связанные между собой с помощью локальной или глобальной сети.

Опишем требования, предъявляемые к архитектуре разрабатываемой системы. Поскольку она является распределенной, эти требования во многом продиктованы требованиями, предъявляемыми к распределенным системам в целом.

Масштабируемость. Система должна быть рассчитана на различные, в том числе большие, нагрузки, а ее архитектура при этом должна оставаться неизменной. Под нагрузкой, в данном случае, будем понимать интенсивность потока поступающих мультимедиа-документов. В зависимости от характеристик конкретной задачи, эта интенсивность может быть значительной. Данное требование, в первую очередь, ведет к необходимости обеспечения возможности одновременной работы нескольких компонентов по обработке мультимедиа-информации.

Открытость. Система должна легко расширяться и модифицироваться. Для этого входящие в нее компоненты должны иметь четко определенные интерфейсы. Система должна быть построена в соответствии с общепринятыми стандартами. Это требование приводит к тому, что для каждого компонента должны быть четко описаны операции, которые он выполняет, а также их параметры. Таким образом, должны быть определены интерфейсы компонентов, которые используются другими компонентами для доступа к функциям обработки.

Прозрачность. Архитектура системы, удовлетворяющая предыдущим требованиям достаточно сложна. В то же время, система должна восприниматься пользователем как единое целое. Это приводит к тому, что факт распределенности системы должен быть максимально скрыт от ее составляющих и пользователей, то есть быть прозрачным. Важна прозрачность местонахождения, означающая максимальную независимость методов идентификации компонентов в рамках данной архитектуры от их реального месторасположения. Требование прозрачности приводит к важному свойству распределенных компонент – прозрачности миграции, означающему, что компоненты могут быть перенесены на другие компьютеры сети без изменения архитектуры системы и так, чтобы эта операция происходила незаметно для процесса обработки. Еще одним типом прозрачности, реализация которого крайне важна, является прозрачность

долговременного хранения. Она состоит в том, что компоненты системы являются долговременными и сохраняют свое состояние на внешнем носителе до тех пор, пока не будут активированы некоторым процессом. В этом случае их состояние считывается.

Отметим, что разрабатываемая архитектура является обобщенной. Конкретная ее реализация может основываться на различных архитектурах создания распределенных систем, таких как DCOM, CORBA, Java/RMI.

3.1. Разработка обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации

С учетом поставленной в первой главе задачи исследования и разработанной в предыдущей главе моделью сущности мультимедиа-информации, методами внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров, методом контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков рассмотрим обобщенную архитектуру программного комплекса контроля целостности мультимедиа-информации. Данная архитектура должна учитывать возможности использования концепции программных агентов и подключения функциональных модулей внедрения ЦВЗ, контроля целостности и т.д. [57, 47]

Очевидно, для использования в современных системах обработки и хранения (организации библиотек и архивов мультимедиа-документов) мультимедиа-информации, построенных на базе электронно-вычислительных сетей, данная архитектура должна иметь распределенный характер с управляющим центром, позволяющим осуществлять оперативное управление системой контроля целостности и ее мониторинг в режиме реального времени. Разрабатываемая архитектура включает в себя типовые блоки внедрения и извлечения ЦВЗ, анализа и контроля целостности, которые могут быть размещены как локально на отдельном узле, так и распределено по защищаемой сети и ее границам.

Согласно разработанной архитектуре комплекс контроля целостности мультимедиа информации состоит из следующих агентов (см. рисунок 18) [46]:

- агент контроля и управления;
- агент информационного хранилища;
- агент управления;
- агент регистрации мультимедиа-информации;

- агент клонирования мультимедиа-информации;
- агент анализа и контроля целостности.

Все агенты, входящие в состав комплекса, построены с использованием модульной архитектуры [67]. В архитектуру всех агентов входит главный программный модуль и функциональные модули. Кроме того, в разработанной архитектуре представлены два типа модулей: обязательные (необходимые для штатной эксплуатации комплекса) и необязательные (наличие которых необязательно для работы комплекса). Таким образом, функциональность работы комплекса может быть изменена либо при помощи подключения новых обязательных модулей, либо при помощи подключения необязательных модулей.

Приведем более подробное описание архитектуры каждого типа агентов.

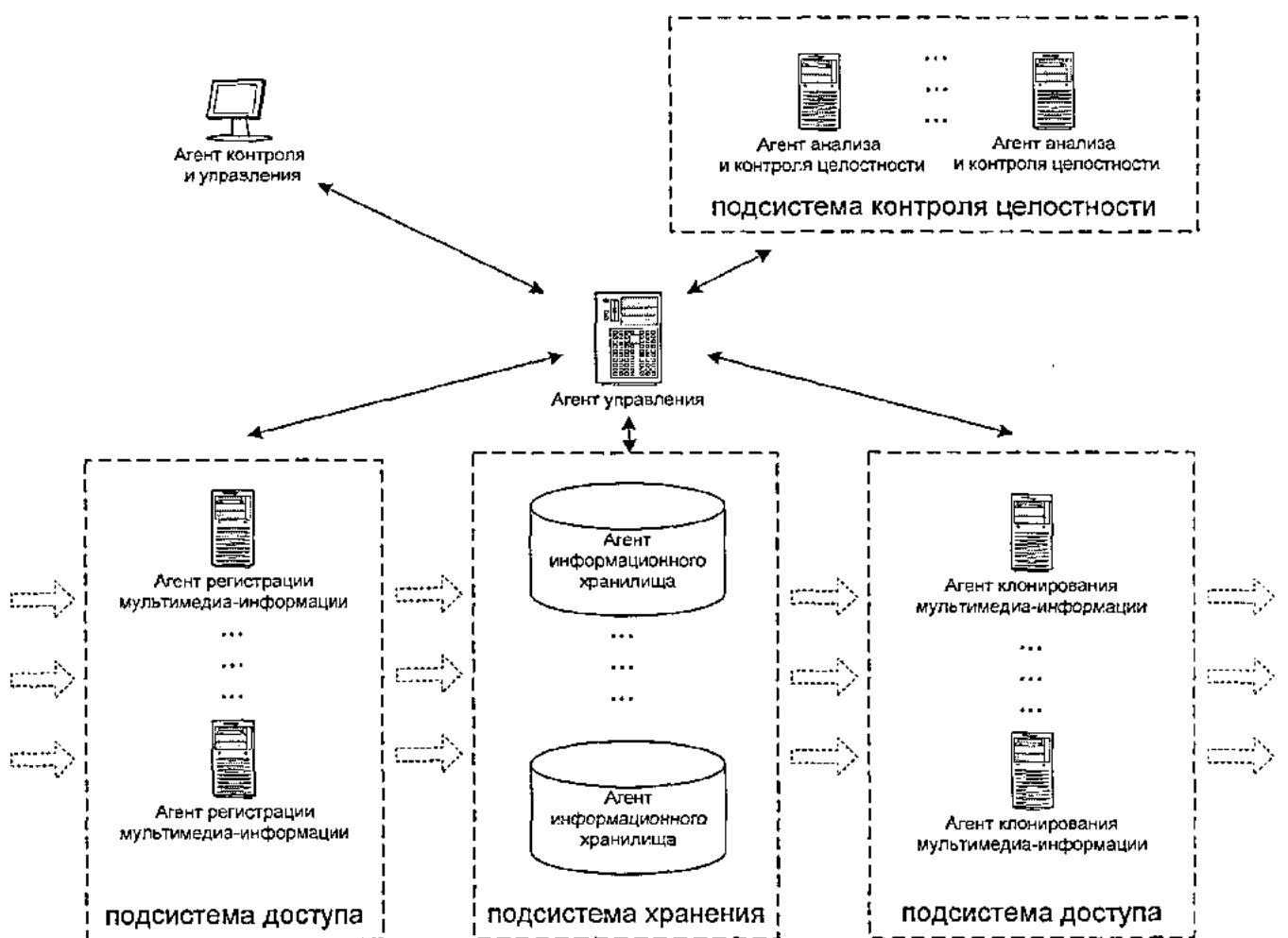


Рисунок 18 – Обобщенная архитектура программного комплекса контроля целостности мультимедиа-информации

3.1.1. Агент контроля и управления

Агент контроля и управления является единым центром управления всем комплексом защиты мультимедиа-информации. Он состоит из двух частей: подсистемы управления и подсистемы оповещения. Подсистема управления позволяет производить настройку структуры комплекса в целом и его конкретных агентов. Подсистема оповещения позволяет осуществлять мониторинг текущего состояния комплекса, включающей информацию о производительности и состоянии агентов, а также о выявленных нарушениях целостности мультимедиа-информации. Кроме того, подсистема оповещения позволяет производить отложенных анализ событий, зафиксированных в журнале информационного хранилища. Агент контроля и управления взаимодействует только с агентом управления. Функциональность данного агента регулируется наборами подключаемых модулей (см. рисунок 19). При этом для корректной работы обязательным является наличие по крайней мере одного подключенного модуля каждого типа.



Рисунок 19 – Архитектура агента контроля и управления

Модуль взаимодействия с агентом управления отвечает за установку соединения с агентом управления, в случае если не получается создать подключение к агенту управления работа данного агента прерывается. После успешной установки соединения данный модуль отвечает за передачу и прием управляющих и информационный воздействий.

Модули управления и взаимодействия с пользователем, являются графическими формами, предназначеными для обеспечения функциональности, и служат для

формирования и отображения информации о работе комплекса. Таким образом они являются посредником между пользователем системы и модулем взаимодействия с агентом управления.

Модули создания отчетов и просмотра журналов, также являются графическими формами. Они позволяют пользователю работать с информацией, хранящейся в информационном хранилище. Их отличие от модулей управления состоит в том, что они не могут влиять на работу комплекса и отдельных его компонентов.

Модули мониторинга состояния и оповещения предназначены для отображения ошибок и предупреждений поступающих от агента управления.

3.1.2. Агент информационного хранилища

Агент информационного хранилища предназначен для организации хранения защищаемой мультимедиа информации, информации о ней, о также служебной информации необходимой для обеспечения работоспособности комплекса. Он взаимодействует с агентом управления и агентами регистрации и клонирования мультимедиа-информации. В архитектуру агента информационного хранилища входят только обязательные модули (см. рисунок 20).

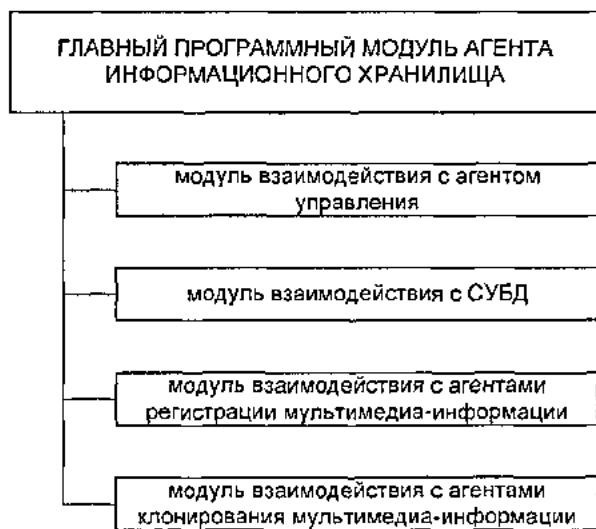


Рисунок 20 – Архитектура агента информационного хранилища

Модуль взаимодействия с агентом управления отвечает за установку соединения с агентом управления, в случае если не получается создать подключение к агенту управления работа данного агента прерывается. После успешной установки соединения

данный модуль отвечает за передачу и прием управляющих и информационный воздействий.

Модуль взаимодействия с СУБД отвечает за установку соединения с используемой СУБД, в случае если не получается создать подключение к СУБД информация об этом направляется агенту управления и работа данного агента прерывается. После успешной установки соединения данный модуль отвечает за формирование SQL запросов, взаимодействие с СУБД, обработку и передачу информации полученной от СУБД.

Модуль взаимодействия с агентами регистрации мультимедиа-информации отвечает за прием соединений от агентов регистрации мультимедиа-информации. После успешной установки соединения с очередным агентом регистрации мультимедиа-информации данный модуль отвечает за обеспечение обмена информацией между СУБД и агентом регистрации.

Модуль взаимодействия с агентами клонирования мультимедиа-информации отвечает за прием соединений от агентов клонирования мультимедиа-информации. После успешной установки соединения с очередным агентом данного модуль отвечает за обеспечение обмена информацией с СУБД.

3.1.3. Агент управления

Агент управления предназначен для управления всеми агентами имеющимися в комплексе и обработки поступающей от них информации. Он взаимодействует со всеми установленными в комплексе агентами. Архитектура данного агента состоит из главного модуля, модулей взаимодействия с различными типами агентов, реализующих установку соединений со всеми агентами, обработку поступающей информации и пересылку управляющих воздействий и модуля журналирования, отвечающего за весь комплекс в целом (см. рисунок 21). Все модули данного агента являются обязательными.

Модули взаимодействия с различными типами агентов отвечают за прием соединений от агентов соответствующих типов. После успешной установки соединения с очередным агентом данные модули отвечают за передачу и прием управляющих и информационный воздействий. На рисунке 22 приведена блок-схема работы модуля взаимодействия с агентом контроля и управления, модули взаимодействия с другими агентами построены аналогично, но они обрабатывают команды не одного подключенного агента, а нескольких агентов одновременно.

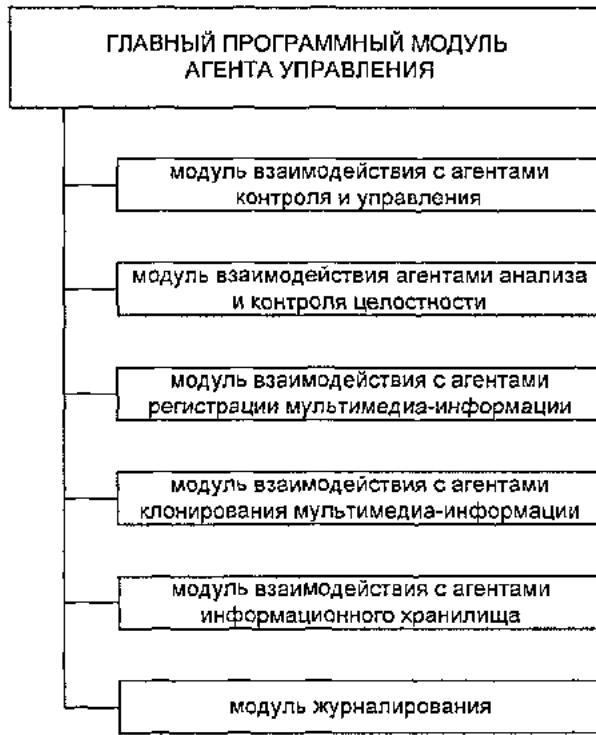


Рисунок 21 – Архитектура агента управления

Модуль журналирования предназначен для обеспечения непрерывного журналирования всех событий происходящих с комплексом. Данный модуль предоставляет интерфейсы, позволяющие главному модулю и модулям взаимодействия осуществлять запись в журнальный файл.

3.1.4. Агент регистрации мультимедиа-информации

Агент регистрации мультимедиа-информации предназначен для добавления мультимедиа-информации в хранилище и внедрения в нее ЦВЗ с кодом контроля целостности и служебной информации. Он взаимодействует с агентом информационного хранилища и агентом управления. Архитектура данного агента состоит из следующих модулей (см. рисунок 23, разрывной линией показано подключение необязательных модулей):

- модуль сокрытия и извлечения ЦВЗ является обязательным и реализует описанные во второй главе методы внедрения и извлечения ЦВЗ;
- модуль выделения пространства сокрытия реализует выделение пространства сокрытия из конкретного типа и формата хранения мультимедиа-информации.

Обязательным для работы агента является наличие хотя бы одного подобного модуля (см. главу 2);

- модуль создания стеганографической ключевой последовательности является обязательным и отвечает за создание ключевой последовательности, необходимой для внедрения и извлечения ЦВЗ (см. главу 2);

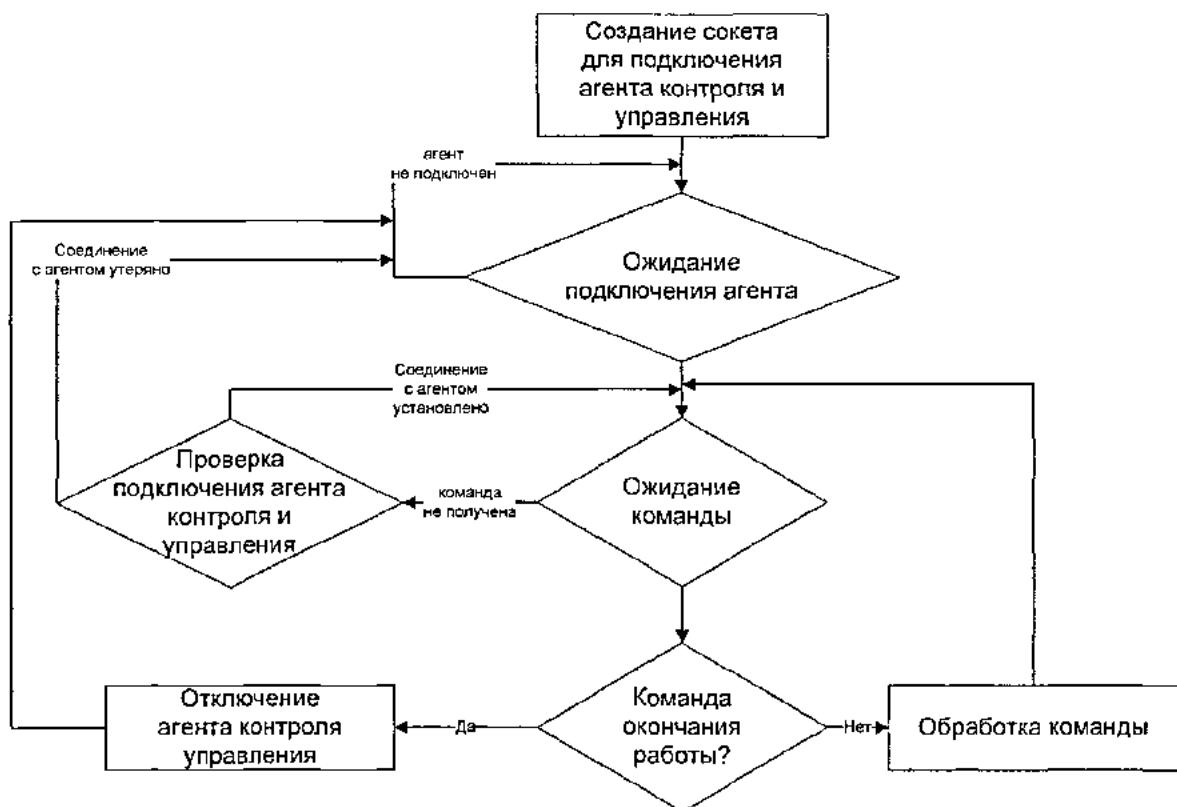


Рисунок 22 – Блок-схема организации взаимодействия с агентом контроля и управления

- модуль контроля целостности является обязательным и реализует разработанный в предыдущей главе метод контроля целостности мультимедиа-информации (см. главу 2);

- модуль шифрования не является обязательным, в случае его наличия выполняется шифрование ЦВЗ перед его внедрением в пространство сокрытия мультимедиа-контейнера;

- модуль работы с источником мультимедиа-информации отвечает за чтение и запись источника (файла, потока и т.п.) мультимедиа-информации. Для работы агента необходимо наличие хотя бы одного подобного модуля (см. главу 2);

- модуль определения естественности не является обязательным и реализует проверку естественности мультимедиа-информации как до, так и после внедрения ЦВЗ согласно разработанной модели естественности (см. главу 2);

- модуль контроля искажений не является обязательным и реализует вычисление искажений в соответствии с разработанным методом контроля искажений (см. главу 2);
- модуль взаимодействия с агентом управления является обязательным и отвечает за установку соединения и обмен информацией с агентом управления;
- модуль взаимодействия с агентом информационного хранилища является обязательным и отвечает за установку соединения с агентом информационного хранилища и добавление в хранилище служебной и мультимедиа-информации.

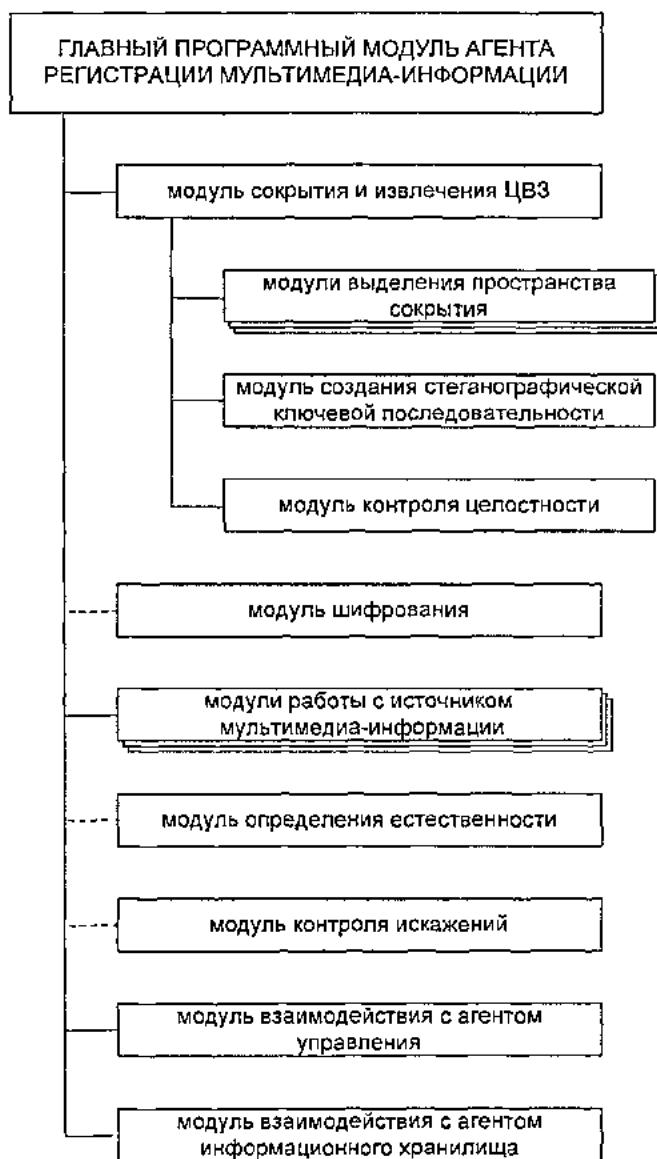


Рисунок 23 – Архитектура агента регистрации мультимедиа информации

3.1.5. Агент клонирования мультимедиа-информации

Агент клонирования мультимедиа-информации предназначен для организации доступа к мультимедиа-информации из информационного хранилища. Его функции состоят в проверке внедренного ЦВЗ и его модификации с целью внедрения идентификатора копии. Он взаимодействует с агентом информационного хранилища и агентом управления. Архитектура данного агента схожа с архитектурой агента регистрации мультимедиа-информации, однако в нем используются модуль взаимодействия с информационным хранилищем, реализующий не добавление, а извлечение мультимедиа-информации (см. рисунок 24).

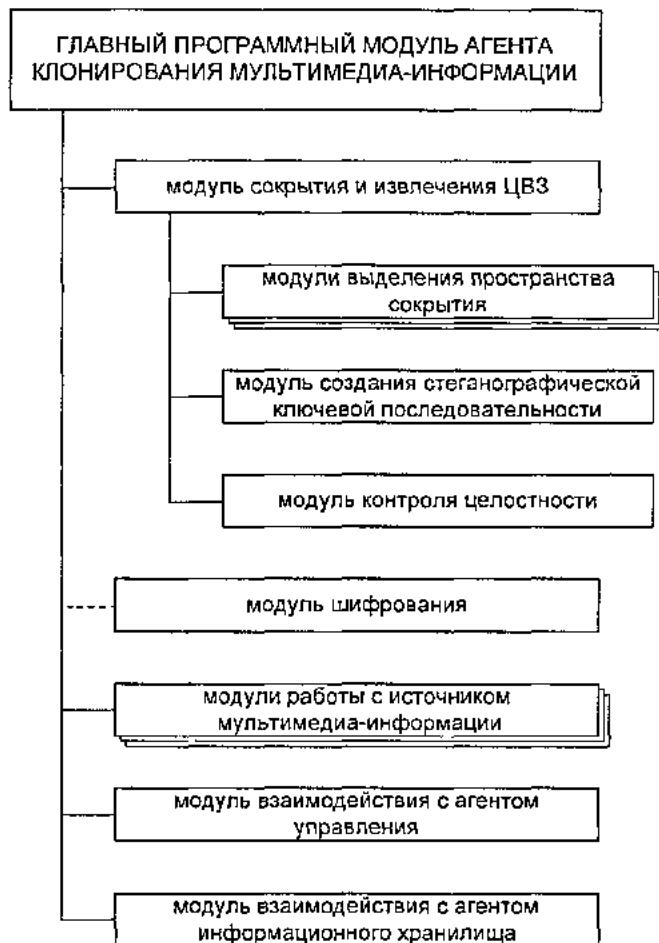


Рисунок 24 – Архитектура агента клонирования мультимедиа-информации

3.1.6. Агент анализа и контроля целостности

Агент анализа и контроля целостности предназначен для анализа мультимедиа-контейнеров как находящихся в информационном хранилище, так и за его пределами (на

локальных и сетевых файловых системах). В его функции входит периодическая проверка целостности и соответствия служебной информации полученной из ЦВЗ информации хранимой в информационном хранилище. Он взаимодействует с агентом информационного хранилища и агентом управления. Архитектура данного агента схожа с архитектурой агента регистрации мультимедиа-информации, однако в нем присутствует обязательный модуль поиска мультимедиа-информации отвечающий за поиск информации, осуществление проверки ее целостности и контроль соответствия сокрытой в ЦВЗ и хранящейся в информационном хранилище информации (см. рисунок 25).

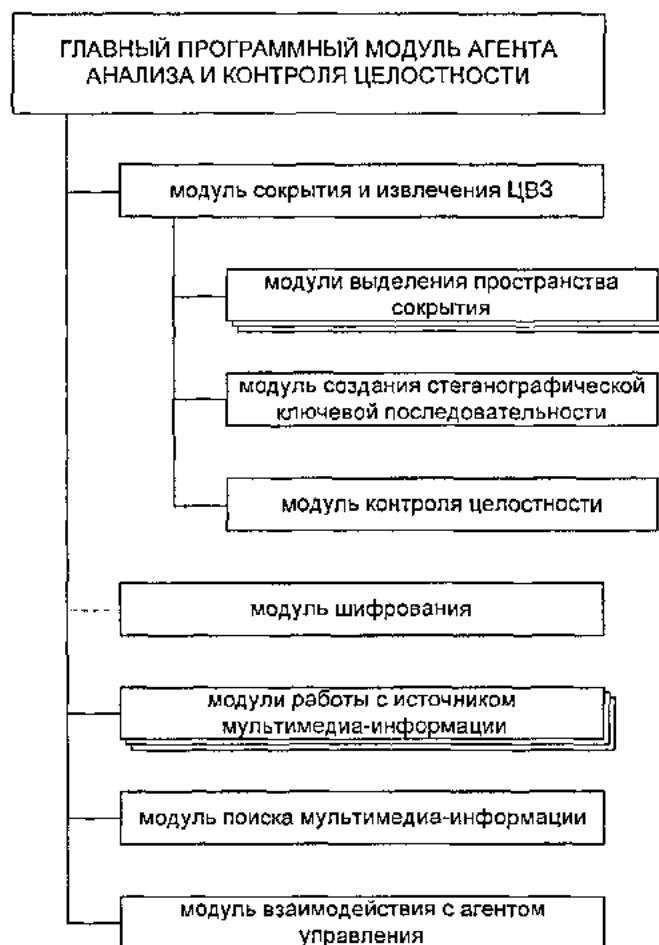


Рисунок 25 – Архитектура агента анализа и контроля целостности

3.2. Разработка универсальных интерфейсов взаимодействия управляющих и функциональных компонент

Для использования алгоритмов, написанных на одном языке, в программах написанных на другом языке используются динамически загружаемые библиотеки (.so файлы на платформе Unix и .dll файлы на платформе Windows).

Универсальные интерфейсы взаимодействия управляющих и функциональных компонент основаны на использовании динамически загружаемые библиотек. Это позволяет регулировать функциональность комплекса, подключая дополнительные модули программного обеспечения.

Интерфейс взаимодействия главного программного модуля с модулем сокрытия и извлечения ЦВЗ состоит из девяти функций следующего вида:

```
typedef void (*exject_t)(short);
typedef int (*inject_t)(int);
typedef void (*in_setup_t)(const char *, uint8_t *, uint16_t, uint8_t);
typedef void (*in_finish_t)();
typedef void (*out_setup_t)(const char *, uint8_t *, uint16_t, uint8_t);
typedef void (*out_finish_t)();
typedef void (*check_setup_t)(const char *, uint8_t *, uint16_t, uint8_t);
typedef CHECKDATA *(*check_result_t)();
typedef void (*check_finish_t)().
```

Интерфейс взаимодействия главного программного модуля с модулем создания стеганографической ключевой последовательности состоит из пяти функций следующего вида:

```
typedef STEG_KEY *keygenopen(uint8_t , uint8_t *, uint16_t);
typedef int (*keyenclose)(STEG_KEY *);
typedef int (*keygenreadbit)(STEG_KEY *);
typedef uint8_t (*keygenreadNbits)(STEG_KEY *, uint8_t);
typedef uint8_t (*keygenreadMask)(STEG_KEY *, uint8_t).
```

Интерфейс взаимодействия главного программного модуля с модулем вычисления кода контроля целостности состоит из двух функций следующего вида:

```
typedef int (*encode_t) (char *data, int data_sz, char **cdata, int
*cdata_sz);
typedef int (*ccheck_t) (char *data, int data_sz, char *cdata, int cdata_sz);
```

Интерфейс взаимодействия главного программного модуля с модулем шифрования состоит из пяти функций следующего вида:

```
typedef CIPHER_KEY *(*shedulekey_t)(char *passphrase, int pass_sz);
typedef int (*encrypt_t)(CIPHER_KEY *key, char *data, int data_sz);
typedef int (*decrypt_t)(CIPHER_KEY *key, char *data, int data_sz);
typedef int (*disposekey_t)(CIPHER_KEY *key);
typedef CIPHER_KEY *(*duplicatekey_t)(CIPHER_KEY *key).
```

Интерфейс взаимодействия главного программного модуля с модулем определения естественности состоит из одной функций следующего вида:

```
typedef bool (*check_natural_t) (char *filename, int filename_sz, double *model_est, int model_sz).
```

Интерфейс взаимодействия главного программного модуля с модулем вычисления искажений состоит из одной функций следующего вида:

```
typedef bool (*check_distort_t) (char *filename_in, int filename_sz_in, char *filename_out, int filename_sz_out, double *model_dist, int model_sz).
```

3.3. Разработка открытых интерфейсов для подключения программного обеспечения стороннего разработчика

Для хранения мультимедиа-информации существует достаточно большое количество всевозможных форматов, кодеков и т.п., многие из которых являются коммерческими продуктами. Это сильно затрудняет получение описания формата и разработку модулей выделения пространства сокрытия. Кроме того, различные алгоритмы кодирования мультимедиа-информации могут эффективно реализовываться на различных языках программирования, в зависимости от наличия уже готовых библиотек. Следовательно, для того, чтобы удовлетворить требованиям к разрабатываемой архитектуре, необходимо обеспечить ее открытыми интерфейсами для подключения программного обеспечения стороннего разработчика.

Для использования алгоритмов, написанных на одном языке, в программах написанных на другом языке используются динамически загружаемые библиотеки (.so файлы на платформе Unix и .dll файлы на платформе Windows). Несмотря на все преимущества данного подхода, он имеет существенный недостаток: при использовании объектно-ориентированных языков программирования он не обеспечивает возможности подключения разделяемых библиотек написанных на одном языке программирования в программу, написанную на другом. В некоторых случаях, даже использование различных компиляторов одного языка делает невозможным использования разделяемых библиотек в программах, собранных другим компилятором.

Фирма Microsoft предложила достаточно эффективное решение этой проблемы – Интерфейсы [122], [127], так называемая СОМ модель (Component Object Model – компонентная модель объектов). Позднее это решение было перенесено и на другие платформы, в том числе Unix. Использование программных интерфейсов позволяет писать

приложения на различных объектно-ориентированных языках, при этом сохраняется возможность использования модулей написанных на одном языке в программах написанных на другом.

Учитывая все вышесказанное, для организации взаимодействия главного программного модуля агентов и модулей выделения пространства сокрытия и работы с форматами хранения мультимедиа-информации был выбран механизм СОМ-интерфейсов.

3.3.1. Интерфейс для подключения модулей для работы с мультимедиа объектами IMediaStream

Концепция потока (Stream) достаточно широко распространена при написании программных продуктов. Она используется в стандарте языка C++ [126, 12], в спецификации СОМ и в другой [129] литературе. Основой этой концепции является представление данных как некоторого цельного потока, из которого можно прочитать порцию данных, записать в него порцию данных, перейти к определенной позиции и так далее. Очевидно, что объекты, используемые в стеганографии (контейнеры, скрываемая информация, пространства сокрытия), хорошо представляются с помощью потоков. Учитывая этот факт, в качестве базовой концепции была выбрана концепция потока.

На рисунке 26 приведено описание интерфейса, описывающего некоторый поток. Опишем назначение функций, составляющих данный интерфейс.

Функции

```
STDMETHOD_( wchar_t*, get_Type)() PURE;  
STDMETHOD( set_Type)(wchar_t* Value) PURE;
```

предназначены для получения/установки типа потока. Для кодирования типа потока используются символьные последовательности следующего вида:
<тип содержимого>/<тип элемента>. Здесь <тип содержимого> может принимать одно из следующих значений: audio, image, video, data. Поле <тип элемента> зависит от поля <тип содержимого> и описывает тип отсчета. Например, возможны следующие типы потоков: image/plain, image/gif, data/byte, audio/plain, audio/alaw, audio/mpeg13, audio/mpeg213.

```
// {0C01A2A7-EAA6-43a8-BD64-4E8865FF66AD}  
DEFINE_GUID( CLSID_CMediaStream,  
0xc01a2a7, 0xea6, 0x43a8, 0xbd, 0x64, 0x4e, 0x88, 0x65, 0xff, 0x66, 0xad);
```

```

// {019EF278-B5BE-4e6e-B28A-F9FE3AC84C37}
DEFINE_GUID( IID_IMediaStream,
             0x19ef278,0xb5be,0x4e6e,0xb2,0x8a,0xf9,0xfe,0x3a,0xc8,0x4c,0x37);

DECLARE_INTERFACE_( IMediaStream, IUnknown ){
public:
    STDMETHOD_( wchar_t*, get_Type)() PURE;
    STDMETHOD_( set_Type)(wchar_t* Value) PURE;
    STDMETHOD_( enumMediaStrmElemType, get_ElementType)() PURE;
    STDMETHOD_( set_ElementType)( enumMediaStrmElemType Value ) PURE;
    STDMETHOD_( int, get_Dims)() PURE;
    STDMETHOD_( set_Dims)( int Value ) PURE;
    STDMETHOD_( enumMediaStrmMode, get_Mode)() PURE;
    STDMETHOD_(      set_Mode)( enumMediaStrmMode Value ) PURE;
    STDMETHOD_( int, get_Size)( int DimIndex ) PURE;
    STDMETHOD_(      set_Size)( int DimIndex, int Value ) PURE;
    STDMETHOD_( int, get_Position)( int DimIndex ) PURE;
    STDMETHOD_(      set_Position)( int DimIndex, int Value ) PURE;
    STDMETHOD_(      Clear)() PURE;
    STDMETHOD_( int, Read )( byte *Buffer, int Size ) PURE;
    STDMETHOD_( int, Write)( byte *Buffer, int Size ) PURE;
    STDMETHOD_( bool, Eos )() PURE;
};


```

Рисунок 26 – Определение интерфейса IMediaStream

Функции

```

STDMETHOD_(enumMediaStrmElemType, get_ElementType)() PURE;
STDMETHOD_( set_ElementType)( enumMediaStrmElemType Value ) PURE;

```

предназначены для получения/установки типа элементов, из которых состоит поток. Они являются зеркальным отражением поля <тип элемента> из типа потока. Список возможных значений определяется перечислением enumMediaStrmElemType и приведен на рисунке 27.

```

typedef enum{
    // потоки необработанных данных
    strmElTData8,

```

```

    strmElTData16,
    strmElTData32,
    strmElTData64,
    // потоки, содержащие аудиоданные
    strmElTAudioALaw,
    strmElTAudioMuLaw,
    strmElTAudio8,
    strmElTAudio16,
    strmElTAudio24,
    strmElTAudio32,
    // потоки, содержащие изображения (для статичных изображений,
размерность
    // равна 2, для видео потоков 3)
    strmElTImageIndex,
    strmElTImageRGB,
    strmElTImageRGBA,
    strmElTImageCMYK,
    // Streams that handle texts
    strmElTTextChar,
    strmElTTextWChar
} enumMediaStrmElemType;

```

Рисунок 27 – Определение возможных типов элементов потока

Функции

```

STDMETHOD_( int, get_Dims)() PURE;
STDMETHOD (sct_Dims)( int Value ) PURE;

```

предназначены для получения/установки количества измерений, присутствующих в потоке. Для аудиосигнала количество измерений всегда равно одному. Для статических изображений, количество измерений равно двум. Для видео последовательностей количество измерений равно трем.

Функции

```

STDMETHOD_(enumMediaStrmMode, get_Mode)() PURE;
STDMETHOD (sct_Mode)( enumMediaStrmMode Value ) PURE;

```

предназначены для получения/установки типа доступа к потоку. Список возможных значений определяется структурой enumMediaStrmMode (см. рисунок 28) и состоит из следующих значений:

strmModeNone – доступ к потоку невозможен;
strmModeRead – доступ к потоку возможен только на чтение;
strmModeWrite – доступ к потоку возможен только на запись;
strmModeReadWrite – доступ к потоку возможен как на чтение так и на запись.

```
typedef enum {  
    strmModeNone,  
    strmModeRead,  
    strmModeWrite,  
    strmModeReadWrite  
} enumMediaStrmMode;
```

Рисунок 28 – Определение типа доступа к элементу потока

Функции

```
STDMETHOD_( int, get_Size)( int DimIndex ) PURE;  
STDMETHOD( set_Size)( int DimIndex, int Value ) PURE;
```

предназначены для получения/установки количества элементов в потоке для каждого измерения. Например, для изображения, имеющего, как было сказано выше, две координаты, первая из них устанавливает размер изображения по вертикали, вторая – по горизонтали. Для видеопоследовательностей первая координата содержит число кадров.

Функции

```
STDMETHOD_( int, get_Position)( int DimIndex ) PURE;  
STDMETHOD( set_Position)( int DimIndex, int Value ) PURE;
```

предназначены для получения/установки номера текущего элемента в потоке, который будет прочитан при следующей операции чтения (или записан при следующей операции записи) для каждого измерения.

Функция

```
STDMETHOD( Clear)() PURE;
```

предназначена для очистки потока и установки размеров по всем измерениям равным нулю.

Функции

```
STDMETHOD_( int, Read )( byte *Buffer, int Size ) PURE;  
STDMETHOD_( int, Write)( byte *Buffer, int Size ) PURE;
```

предназначены для чтения/записи элементов, на которые указывает переменная Buffer в поток. Количество элементов задается переменной Size. Функции не контролируют конец измерения. Если достигнут конец младшего измерения, то увеличивается позиция в старшем измерении и чтение/запись продолжается.

Функция

```
STDMETHOD_( bool, Eos )() PURE;
```

предназначена для получения информации о том, достигнут конец потока или нет.

3.3.2. Интерфейс для подключения модулей выделения пространства скрытия IMediaContainer

Одним из основных понятий в стеганографии является понятие контейнера. Для определения понятия контейнера в программном интерфейсе был введен интерфейс IMediaContainer (см. рисунок 29). Как видно из определения, под контейнером понимается объект, обладающий некоторыми свойствами и имеющий один входной поток и несколько выходных. Например, контейнер, реализующий работу с Wave файлом, имеет в качестве входного потока поток, реализующий работу с файлом на диске, а в качестве выходных потоков выступают аудиосигналы, соответствующие различным каналам. Т.е. для моно сигнала выходной поток будет один, для стерео сигнала количество выходных потоков будет равно двум. Заметим, что термин "входной поток" в данном случае не означает, что этот поток используется только для чтения. Например, в случае создания заполненного контейнера, входной поток играет роль выходного, поскольку в него данные будут записываться. Использование терминов справедливых для чтения контейнеров и инвертирующихся при записи контейнера было выбрано, для облегчения общей модели классов и интерфейсов и упрощения их реализации.

```
typedef enum{
    mcPropIVersion,      /* int */
    mcPropName,          /* wchar_t* */
    mcPropSource         /* wchar_t* | NULL */
} enumMediaCProp;

// {8C47F55E-BCC5-4499-BAFC-EE5F13F04192}
DEFINE_GUID( IID_IMediaContainer,
    0x8c47f55e, 0xbcc5, 0x4499, 0xba, 0xfc, 0xee, 0x5f, 0x13, 0xf0, 0x41, 0x92);
```

```

// {ACF53EFA-DC6E-4b99-9456-A4AE8A74F008}
DEFINE_GUID( CLSID_CMediaContainer,
    0xacf53efa, 0xdc6e, 0x4b99, 0x94, 0x56, 0xa4, 0xae, 0x8a, 0x74, 0xf0, 0x8);

DECLARE_INTERFACE_( IMediaContainer, IUnknown ) {
public:
    STDMETHOD_( IMediaStream*, get_InStream)() PURE;
    STDMETHOD( set_InStream)(IMediaStream* Value) PURE;
    STDMETHOD_( IArrayOfMediaStream*, get_OutStreams)() PURE;
    STDMETHOD( get_Property)(enumMediaCProp PropId, void**PropValue ) PURE;
    STDMETHOD( set_Property)(enumMediaCProp PropId, void* PropValue ) PURE;
};


```

Рисунок 29 – Определение интерфейса IMediaContainer

Опишем назначение функций, составляющих данный интерфейс.

Функции

STDMETHOD_(IMediaStream*, get_InStream)() PURE;

STDMETHOD(set_InStream)(IMediaStream* Value) PURE;

предназначены для получения/установки "входного потока". Функция set_InStream должна понимать Value равное NULL, которое означает, что входного потока нет. Данная возможность используется для повышения стабильности работы программы - перед завершением работы, чтобы не было обращений по несуществующим указателям.

Функция

STDMETHOD_(IArrayOfMediaStream*, get_OutStreams)() PURE;

предназначена для получения доступа к массиву "выходных" потоков. Она возвращает указатель на интерфейс IArrayOfMediaStream, используя который можно получить доступ к любому выходному потоку, а также выполнять некоторые действия с ними. Более подробно этот интерфейс описан далее.

Функции

STDMETHOD(get_Property)(enumMediaCProp PropId, void**PropValue) PURE;

STDMETHOD(set_Property)(enumMediaCProp PropId, void* PropValue) PURE;

предназначены для чтения/установки различных свойств контейнера. Номер устанавливаемого/читаемого свойства задается параметров PropId и определяется перечислением enumMediaCProp (см. рисунок 29). Доступны следующие свойства:

mcPropIVersion – число, указывающее версию интерфейсов, поддерживаемых контейнером;

mcPropName – строка, задающая имя контейнера;

mcPropSource – используется контейнером, реализующим работы с файлом, для установки имени файла, который необходимо читать/записывать.

Существует альтернативный механизм реализации данных функций, когда для каждого свойства создается своя пара функций чтения/установки. По сути, это означает, что в реализованном подходе связывание переносится на этап выполнения, а в случае введения специальных функций на этап компиляции. Каждый подход имеет свои преимущества и недостатки. При использовании для каждого свойства специальных функций чтения/установки повышается скорость выполнения, но введение нового свойства приводит к необходимости изменения интерфейса, что требует внесения изменений во все созданные функциональные модули. С другой стороны не все функциональные модули должны поддерживать все свойства. Например, свойство mcPropSource имеет смысл лишь для функционального модуля, реализующего доступ к файлам. Также может потребоваться введение новых свойств, не влияющих на функциональность в случае, если они не реализованы. Например, возможно введение свойства mcPropLongDesc – описывающего свойства контейнера. Учитывая вышеизложенное, в ситуациях, когда интерфейс стабилен и требуется максимальная скорость доступа к различным свойствам – наилучшим выходом является реализация отдельных функций чтения/установки для каждого свойства. В случае, когда не требуется максимальная скорость доступа, или интерфейс предполагается расширять и дополнять специфическими функциями для конкретных функциональных модулей наилучшим выходом является перенос связывания на этап выполнения путем введения всего двух функций чтения/установки.

Подход, при котором функции чтения/установки реализуются отдельно, для каждого свойства был использован в интерфейсе IMediaStream. Подход, при котором функции чтения/установки реализуются для всех свойства одними и теми же функциями, был использован в интерфейсе IMediaContainer.

3.3.3. Интерфейс для подключения модулей работы с массивами мультимедиа потоков IArrayOfMediaStream

При описании интерфейса IMediaContainer был использован интерфейс, реализующий работу с массивом потоков – IArrayOfMediaStream. Определение интерфейса IArrayOfMediaStream приведено на рисунке 30. Из определения видно, что данный интерфейс позволяет получить доступ к любому потоку по его порядковому номеру, получить количество потоков, а также выполнять операции над потоками, такие как добавление и удаление.

```
// {335EC1A8-DE98-40eb-BC39-59D5F0602712}
DEFINE_GUID( IID_IArrayOfMediaStream,
    0x335ec1a8, 0xde98, 0x40eb, 0xbc, 0x39, 0x59, 0xd5, 0xf0, 0x60, 0x27, 0x12);
DECLARE_INTERFACE_( IArrayOfMediaStream, IUnknown) { //IArray: IList,
ICollection
    STDMETHOD_( IMediaStream*, get_Item)( int Index ) PURE;
    STDMETHOD_( int, get_Count)() PURE;
    STDMETHOD( Insert )( int Index, IMediaStream * Value ) PURE;
    STDMETHOD( Add ) ( IMediaStream * Value ) PURE;
    STDMETHOD( RemoveAt)( int Index ) PURE;
    STDMETHOD( Clear ) () PURE;
};

};
```

Рисунок 30 – Определение интерфейса IArrayOfMediaStream

Опишем каждую функцию более подробно.

Функция

STDMETHOD_(IMediaStream*, get_Item)(int Index) PURE;

предназначена для получения доступа к потоку по его порядковому номеру. Нумерация потоков начинается с нуля. В случае если номер больше существующего числа потоков или меньше нуля возвращается NULL.

Функция

STDMETHOD_(int, get_Count)() PURE;

предназначена для получения числа потоков в данном массиве.

Функция

STDMETHOD(Insert)(int Index, IMediaStream * Value) PURE;

предназначена для добавления нового потока в заданную позицию. Все существующие потоки в начиная с данной позиции будут сдвинуты на одну позицию в сторону старших номеров.

Функция

STDMETHOD(Add)(IMediaStream * Value) PURE;

предназначена для добавления нового потока в конец массива.

Функции Insert и Add должны проверять поддерживает ли контейнер тип, который им передается (по полям Type, ElementType) и если не поддерживается возвращать E_NOTIMPL.

Функция

STDMETHOD(RemoveAt)(int Index) PURE;

предназначена для удаления потока, заданного индексом из массива.

Функция

STDMETHOD(Clear)() PURE;

предназначена для очистки массива. При этом удаляются все потоки.

Использование технологий СОМ предполагает также и использование так называемых "фабрик" для создания конкретных экземпляров классов – объектов. Для этого существует стандартный интерфейс IClassFactory, позволяющий с помощью функции CreateInstance создавать объекты. Поскольку агенты комплекса построены по модульной архитектуре, то необходимо иметь возможность получать описания создаваемых фабрикой объектов, без необходимости создания самих объектов. Для решения этой задачи был введен интерфейс IMediaContainerFactory, унаследованный от IClassFactory, и расширяющий его функциональность (см рисунок 31). В интерфейс была добавлена функция get_Property, позволяющая производить чтение различных свойств, описывающих контейнеры, создаваемые данным подключаемым модулем.

Список доступных свойств задается перечислением enumMediaCFProp и состоит из следующих значений:

mcfPropIVersion – число, указывающее версию интерфейсов, поддерживаемых контейнером.

mcfPropName – строка, задающая имя контейнеров, создаваемых данной фабрикой.

mcfPropShortDesc – строка, содержащая краткое описание функций выполняемых контейнерами, создаваемыми данной фабрикой.

mcfPropLongDesc – строка, содержащая краткое описание функций выполняемых контейнерами, создаваемыми данной фабрикой.

mcfPropCanRecognizeMedia – булево значение, говорящее о том, способны ли контейнеры, создаваемые данной фабрикой самостоятельно распознавать во входном потоке, поток, поддерживаемый ими. Например, контейнер, реализующий чтение/запись к Wave файлам, способен распознать свой поток на основе анализа сигнатур.

mcfPropCanUseInStream – булево значение, говорящее о том, способны ли контейнеры, создаваемые данной фабрикой использовать входной поток, если он установлен. Например, контейнер, реализующий доступ к файлам, не использует входной поток.

mcfPropInStreamTypes – строка, содержащая типы входных потоков, поддерживаемых данным контейнером.

mcfPropOutStreamTypes – строка, содержащая типы выходных потоков, поддерживаемых данным контейнером.

```
typedef enum{
    mcfPropIVersion,           /* int */
    mcfPropName,               /* wchar_t* */
    mcfPropShortDesc,          /* wchar_t* */
    mcfPropLongDesc,           /* wchar_t* */
    mcfPropCanRecognizeMedia,  /* bool */
    mcfPropCanUseInStream,     /* bool */
    mcfPropInStreamTypes,      /* wchar_t* */
    mcfPropOutStreamTypes      /* wchar_t* */
} enumMediaCFProp;

// {1C501768-BBE6-4956-8A34-B66496955EAD}
DEFINE_GUID( IID_IMediaContainerFactory,
    0x1c501768, 0xbbbe6, 0x4956, 0x8a, 0x34, 0xb6, 0x64, 0x96, 0x95, 0x5e, 0xad);

// {9F9BD07E-65E4-4524-AE51-F8A0C2B53DCF}
DEFINE_GUID( CLSID_CMediaContainerFactory,
    0x9f9bd07e, 0x65e4, 0x4524, 0xa8, 0x51, 0xf8, 0xa0, 0xc2, 0xb5, 0x3d, 0xcf);

DECLARE_INTERFACE_( IMediaContainerFactory, IClassFactory ) {
public:
```

```
STDMETHOD( get_Property )( enumMediaCFProp PropId, void ** PropValue )
PURE;
};
```

Рисунок 31 – Определение интерфейса IMediaContainerFactory

Использование разделяемых библиотек при использовании концепции COM подразумевает реализацию в них следующих функций:

```
HRESULT STDAPICALLTYPE DllCanUnloadNowFunc();
HRESULT STDAPICALLTYPE DllGetClassObjectFunc(
    REFCLSID rclsid, REFIID riid, LPVOID *ppv);
HRESULT STDAPICALLTYPE DllRegisterServerFunc();
HRESULT STDAPICALLTYPE DllUnRegisterServerFunc();
```

Назначение этих функций широко описано в литературе, например в [122], [127].

Для облегчения реализации этих функций введен ряд определений, таких, что определение всех необходимых для функционирования комплекса функций имеет вид, приведенный на рисунке 32.

```
MY_COM_DLL_STUB(
    MY_COM_DLL_STUB_ENTRY( CLSID_CMediaContainer, CCont_Factory<CContainer>
)
)
```

Рисунок 32 – Определение экспортируемых из разделяемой библиотеки функций

3.4. Выводы

Для эффективной реализации и использования методов и моделей, разработанных во второй главе, необходимым условием является создание архитектуры программного комплекса, которая с одной стороны должна учитывать особенности разработанных методов, а с другой стороны позволять преодолеть выявленные недостатки существующих систем защиты и контроля целостности. Решению данной частной научной задачи посвящена данная глава.

1. В данной главе сформулированы требования, предъявляемые к разрабатываемой архитектуре, уточняющие формулировку соответствующей частной задачи исследования. Для разработки обобщенной архитектуры был выбран мультиагентный подход, позволяющий использовать разработанные методы внедрения и извлечения ЦВЗ и контроля целостности в рамках единой информационно-вычислительной сети. Основными требованиями, предъявляемыми к разрабатываемой архитектуре, являются масштабируемость, открытость и прозрачность.

2. В данной главе была разработана обобщенная архитектура программного комплекса контроля целостности мультимедиа-информации. Данная архитектура может быть реализована на различных архитектурах создания распределенных систем, таких как DCOM, CORBA, Java/RMI. За счет использования специальных агентов, данная архитектура позволяет осуществлять гибкое масштабирование комплекса и его настройку в соответствии с уже имеющейся распределенной информационно-вычислительной инфраструктурой.

3. Разработанная архитектура состоит из программных агентов шести типов: агент контроля и управления представляет собой единый центр управления всем комплексом защиты мультимедиа-информации, агент информационного хранилища предназначен для организации распределенного хранения служебной и мультимедийной информации, агент управления предназначен для управления всеми агентами комплекса, агент регистрации мультимедиа-информации предназначен для внедрения ЦВЗ с кодом контроля целостности, агент клонирования мультимедиа-информации предназначен для предоставления пользователям копий мультимедиа-информации из хранилища с уникальным ЦВЗ, агент анализа и контроля целостности предназначен для поиска мультимедиа-информации, извлечения ЦВЗ, проверки его корректности и контроля целостности.

4. По результатам первых двух глав исследования даны результаты разработки архитектуры агентов всех типов. Все агенты, входящие в состав комплекса, построены с использованием модульной архитектуры. В архитектуру всех агентов входит главный программный модуль и функциональные модули, которые могут быть либо обязательными (необходимыми для штатной эксплуатации комплекса), либо необязательными (наличие которых необязательно для работы комплекса). Подобная архитектура агентов позволяет быстро изменять и наращивать их функциональность за счет разработки и подключения новых модулей.

5. Описаны универсальные интерфейсы взаимодействия управляющих и функциональных компонент агентов комплекса. Данные интерфейсы используют технологию динамически подключаемых библиотек. Их универсальность заключается в том, что они позволяют, снизить сложность разработки агентов за счет подключения одних и тех же модулей к различным агентам. Кроме того, использование данных интерфейсов позволяет производить изменение используемых в агентах функциональных модулей непосредственно во время работы комплекса.

6. Разработаны и описаны открытые интерфейсы для подключения программного обеспечения стороннего разработчика, а именно интерфейс для подключения модулей для работы с мультимедиа объектами IMediaStream, интерфейс для подключения модулей выделения пространства сокрытия IMediaContainer, интерфейс для подключения модулей работы с массивами мультимедиа потоков IArrayOfMediaStream. Данные интерфейсы позволяют подключать модули, отвечающие за выделение пространства сокрытия и работу с различными форматами хранения мультимедиа-информации. Они построены при помощи механизма СОМ интерфейсов, что позволяет упростить процесс разработки и отладки модулей сторонними разработчиками.

Глава 4. Практические аспекты реализации программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков

Данная глава посвящена практическим аспектам реализации программного комплекса контроля целостности мультимедиа-информации [79]. В ней приводятся особенности реализации главного модуля программного комплекса, а также программного обеспечения функциональных компонент, реализующих работу с мультимедиа-контейнерами, сокрытие и извлечение цифровых водяных знаков и контроль целостности. Проводится оценка вносимых искажений и нарушения естественности мультимедиа-контейнеров и эффективности по скорости, памяти, вычислительным ресурсам, вносимым искажениям и нарушению естественности мультимедиа-контейнеров.

На основе разработанной обобщенной архитектуры программного комплекса контроля целостности мультимедиа-информации для демонстрации возможности реализации предлагаемых принципов и архитектурных решений был разработан программный комплекс информационной защиты мультимедиа-информации с использованием цифровых водяных знаков [71, 72, 109].

Для разработки данного программного комплекса использовалась платформа Microsoft Windows XP/2000/2003. Для разработки главных программных модулей агентов, содержащих элементы графического интерфейса, использовалась среда разработки Borland C++ Builder 6, функциональные модули реализовывались также на языке программирования C++ с использованием как среды разработки Borland C++ Builder 6, так и Microsoft Visual Studio .Net. В процессе разработки использовалась стандартная библиотека языка C++ STL (Standard Template Library). Для реализации интерфейсов подключения модулей сторонних разработчиков к главным программным модулям агентов комплекса была использована библиотека DCOM (Distributed COM) и библиотека ATL (Active Template Library), входящая в состав стандартных библиотек Microsoft Visual C++ [122, 127].

В качестве хранилища данных система использует СУБД Microsoft SQL Server 2000. При реализации функциональных модулей взаимодействия с СУБД доступ к СУБД Microsoft SQL Server 2000 осуществлялся посредством интерфейса OLE DB и библиотеки ATL [127, 126, 12].

4.1. Реализация программного обеспечения функциональных компонент, реализующих работу с мультимедиа-контейнерами, сокрытие и извлечение цифровых водяных знаков и контроль целостности

Разработанный комплекс основан на используемом в обобщенной архитектуре мультиагентном подходе и состоит из следующих программных агентов (см. рисунок 33):

- агент контроля и управления;
- агент управления и организации хранения;
- агент регистрации мультимедиа-информации;
- агент анализа и контроля целостности.

Агенты имеют модульную архитектуру (см. Главу 3) и реализованы с использованием среды разработки Borland C++ Builder 6. Функциональные модули реализовывались также на языке программирования C++ с использованием как среды разработки Borland C++ Builder 6, так и Microsoft Visual Studio .Net.

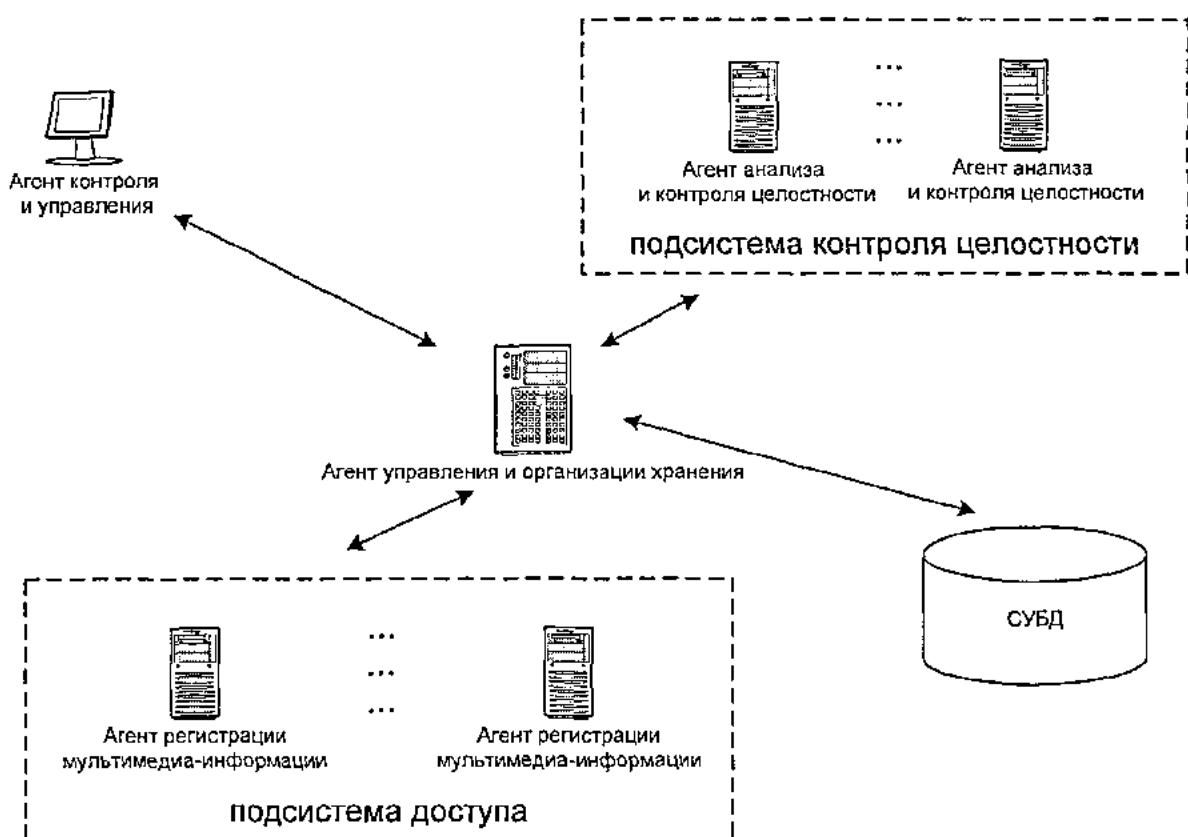


Рисунок 33 – Архитектура программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков

4.1.1. Реализация функциональных модулей, использующих универсальные интерфейсы взаимодействия управляющих и функциональных компонент

В соответствии с разработанной архитектурой программного комплекса в его состав входят следующие функциональные модули, использующие универсальные интерфейсы взаимодействия с главным программными модулями агентов:

- модуль шифрования (sva_crypto.dll) – необязательный;
- модуль кодов контроля целостности (sva_ecc.dll) – обязательный;
- модуль определения естественности (sva_natural.dll) – необязательный;
- модуль оценки искажений (sva_distort.dll) – необязательный;
- модуль сокрытия и извлечения ЦВЗ (sva_stego.dll) – обязательный;
- модуль создания стеганографической ключевой последовательности (sva_keygen.dll) – обязательный.

Загрузка функциональных модулей осуществляется главными программными модулями агентов. На рисунке 34 приведена блок-схема работы главного программного модуля агента управления, работа других главных модулей отличается отсутствием установления связи СУБД и сопутствующих действий.

Модуль шифрования

Данный модуль является необязательным и предназначен для обеспечения криптографической защиты внедряемого ЦВЗ. Согласно разработанному алгоритму сокрытия ЦВЗ функции данного модуля используются на этапе подготовки ЦВЗ к сокрытию и на этапе извлечения ЦВЗ. Отметим, что на приемной и на передающей стороне должен быть подключен один и тот же модуль шифрования, иначе извлечение сокрытого ЦВЗ будет невозможно.

Основной программный модуль пытается загрузить модуль шифрования из файла sva_crypto.dll. В случае неудачной загрузки работоспособность основного программного модуля нарушена не будет. Однако, шифрование ЦВЗ перед сокрытием является желательным и должно применяться всегда, когда его применение возможно.

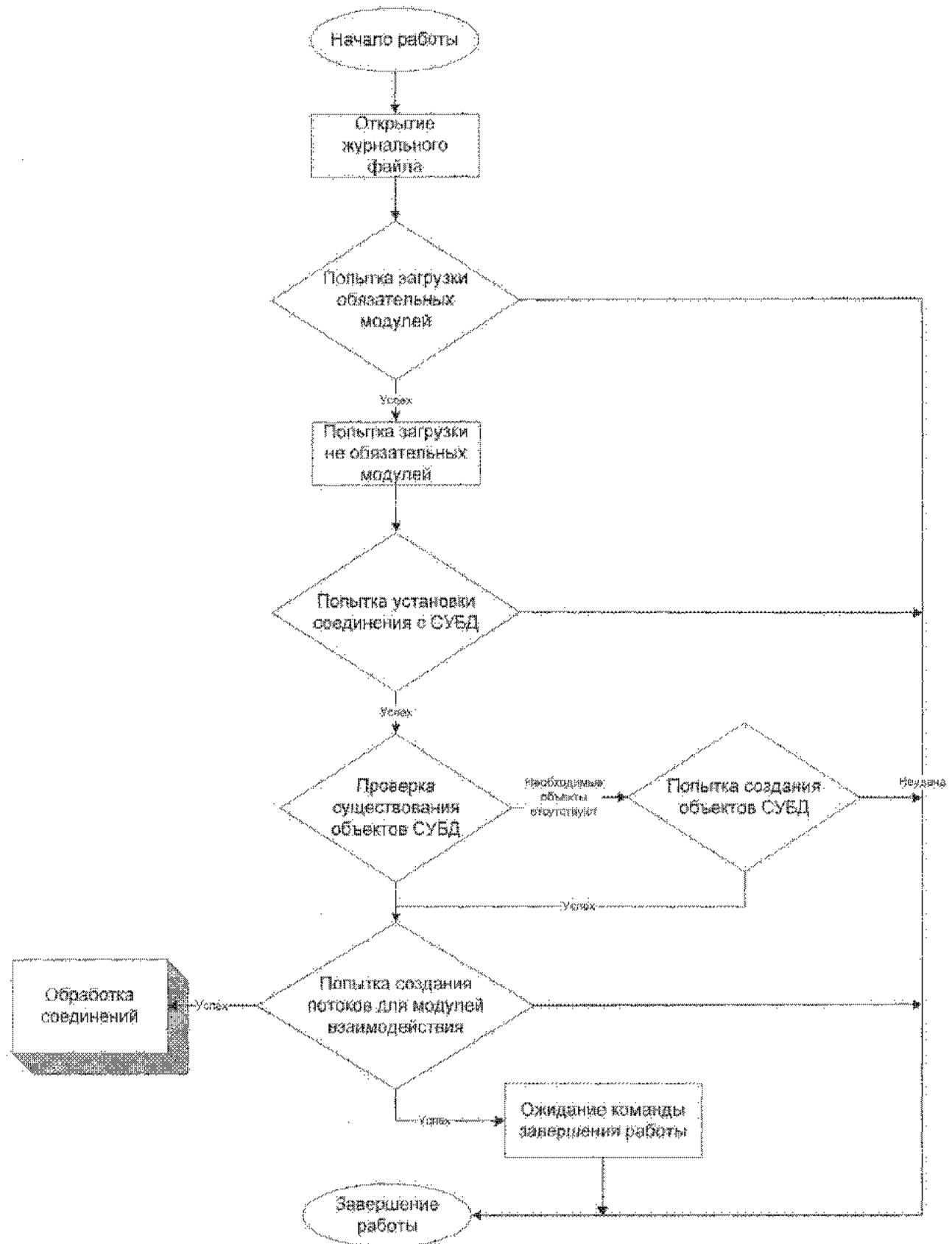


Рисунок 34 – Блок-схема работы главного программного модуля агента управления

Функциональность данного модуля состоит в выполнении трех основных функций: генерации ключа шифрования на основе парольной фразы шифрования, введенной пользователем, шифрование данных сообщения с использованием ключа шифрования и расшифрование данных сообщения с использованием ключа шифрования. В качестве алгоритма шифрования может быть выбран произвольный криптографически стойкий симметричный шифр.

Для обеспечения гибкости в использовании различных алгоритмов шифрования для подключения этого модуля был разработан специальный интерфейс, состоящий из следующих функций: `shedulekey`, `encrypt`, `decrypt`, `disposkey` и структуры `CIPHER_KEY`. Рассмотрим назначение функций и структур подробнее.

Структура `CIPHER_KEY` является универсальным представлением ключевых данных произвольной криптосистемы. Имеет произвольную внутреннюю структуру, которая зависит от конкретного алгоритма:

```
typedef char CIPHER_KEY;
```

Функция `shedulekey` разворачивает ключ шифрования из заданной парольной фразы шифрования и возвращает структуру `CIPHER_KEY`:

```
CIPHER_KEY *shedulekey(char *passphrase, int pass_sz), где
```

`passphrase` – парольная фраза шифрования (или ключевые бинарные данные);

`pass_sz` – длина парольной фразы.

Функция `encrypt` шифрует данные `data` размера `data_sz` с текущим ключом шифрования `key` и возвращает `SUCCESS` или `FAILURE`, в случае удачного и неудачного завершения работы соответственно:

```
int encrypt(CIPHER_KEY *key, char *data, int data_sz), где
```

`key` – текущие данные ключа системы;

`data` – данные для зашифрования;

`data_sz` – размер данных.

Функция `decrypt` расшифровывает данные `data` размера `data_sz` с текущим ключом шифрования `key` и возвращает `SUCCESS` или `FAILURE`:

```
int decrypt(CIPHER_KEY *key, char *data, int data_sz), где
```

`key` – текущие данные ключа системы;

`data` – данные для расшифрования;

`data_sz` – размер данных.

Функция `disposekey` предназначена для завершения процесса шифрования и уничтожения структуры `key` и возвращает `SUCCESS` или `FAILURE`:

```
int disposekey(CIPHER_KEY *key), где  
key – текущие данные ключа системы.
```

Ее необходимо вызывать только после окончания работы с криптосистемой на данном ключе `key`.

Модуль определения естественности

Данный модуль является необязательным и предназначен для определения естественности мультимедиа-информации в соответствии с разработанной моделью естественности. Функции данного модуля используются до и после сокрытия ЦВЗ. Даже если приемной и на передающей стороне будут подключены различные модули определения естественности, не зависимо от этого будет возможно извлечение сокрытого сообщения.

Основной программный модуль пытается загрузить его из файла `sva_natural.dll` и в случае неудачной загрузки сохраняет свою работоспособность. Применение данного модуля повышает стойкость стеганографической системы к атакам пассивного противника.

Интерфейс данного модуля содержит одну функцию `check_natural`, возвращающую вероятность естественности мультимедиа-информации заключенную в пределах от 0 до 1 (естественная мультимедиа-информация, пригодная для сокрытия ЦВЗ):

```
double check_natural (char *filename, int filename_sz), где  
filename – строка, содержащая полный путь к файлу с мультимедиа-контейнером,  
filename_sz – размер строки.
```

Модуль сокрытия и извлечения ЦВЗ

Данный модуль является обязательным и предназначен для осуществления сокрытия и извлечения ЦВЗ, согласно разработанным методам, а также для определения размера пространства сокрытия.

Основной программный модуль пытается загрузить его из файла `sva_stego.dll`, в случае неудачной загрузки дальнейшая работа программного комплекса невозможна.

Перечислим функции и структуры данного модуля, которые используются в работе основного программного модуля:

Структура CHECK_DATA содержит характеристики пространства сокрытия.

Функция `in_setup` осуществляет инициализацию процесса сокрытия ЦВЗ в пространство сокрытия:

`void in_setup(uint8_t *passwd, uint16_t passwdlen, uint8_t m_size)`, где

`passwd` – стеганографическая парольная фраза, используемая для сокрытия информации и определяющая способ сокрытия ЦВЗ. Отметим, что для извлечения сокрытого ЦВЗ необходима парольная фраза, использованная при его сокрытии;

`passwdlen` – длина парольной фразы;

`m_size` – размер блока стеганографической ключевой последовательности, определяющий плотность сокрытия информации, принимает значения от 1 (максимальная плотность сокрытия) до 255 (минимальная плотность сокрытия) (см. таблицу 2).

Функция `in_finish` предназначена для завершения процесса сокрытия и освобождении используемых ресурсов и памяти:

`void in_finish()`.

Функция `in_setup` осуществляет инициализацию процесса извлечения ЦВЗ из пространства сокрытия:

`void out_setup(uint8_t *passwd, uint16_t passwdlen, uint8_t m_size)`, где

`passwd` – стеганографическая парольная фраза, используемая для извлечения ЦВЗ и определяющая способ извлечения ЦВЗ. Отметим, что для извлечения ЦВЗ необходима парольная фраза использованная при его сокрытии;

`passwdlen` – длина парольной фразы;

`m_size` – размер блока стеганографической ключевой последовательности, определяющий плотность сокрытия информации, принимает значения от 1 (максимальная плотность сокрытия) до 255 (минимальная плотность сокрытия) (см. таблицу 2).

Функция `out_finish` предназначена для завершения процесса извлечения и освобождения используемых ресурсов и памяти:

`void out_finish()`.

Функция `check_setup` осуществляет инициализацию процесса определения размера пространства сокрытия, которое предполагается использовать для сокрытия информации:

`void check_setup(uint8_t *passwd, uint16_t passwdlen, uint8_t m_size)`, где

`passwd` – стеганографическая парольная фраза, используемая для сокрытия и извлечения ЦВЗ, однозначно определяющая пространство сокрытия. Отметим, что для сокрытия и извлечения ЦВЗ необходимо использовать ту же парольную фразу, иначе размер пространства сокрытия может отличаться от вычисленного;

`passwdlen` – длина парольной фразы;

`m_size` – размер блока стеганографической ключевой последовательности, определяющий плотность сокрытия информации, принимает значения от 1 (максимальная плотность сокрытия) до 255 (минимальная плотность сокрытия). При определении размера сокрытия, сокрытии и извлечении должен использоваться один и тот же размер блока стеганографической ключевой последовательности (см. таблицу 2).

Функция `check_result` предназначена для получения размера пространства сокрытия:

```
CHECK_DATA * check_result();
```

Функция `out_finish` предназначена для завершения процесса определения размера пространства сокрытия и освобождения используемых ресурсов и памяти:

```
void check_finish();
```

Функция `inject` предназначена для сокрытия очередного бита ЦВЗ в элементе пространства сокрытия `intval`. Ее результатом является измененный элемент пространства сокрытия:

```
int inject (int intval), где
```

`intval` – элемент пространства сокрытия.

Функция `exject` предназначена для извлечения очередного бита сокрытого ЦВЗ из элемента пространства сокрытия `outval`:

```
void exject(short outval), где
```

`outval` – элемент пространства сокрытия.

Модуль создания стеганографической ключевой последовательности

Данный модуль является обязательным и предназначен для создания стеганографической ключевой последовательности, которая задает способ сокрытия ЦВЗ в пространстве сокрытия контейнера. Отметим, что на приемной и на передающей стороне должен быть подключен один и тот же модуль создания стеганографической ключевой последовательности, иначе извлечение сокрытого сообщения будет невозможно.

Модуль сокрытия и извлечения информации пытается загрузить его из файла sva_keygen.dll и в случае неудачной загрузки, работа соответствующего агента программного комплекса будет прервана.

Таблица 2 – Влияние размера блока стеганографической ключевой последовательности на размер используемого пространства сокрытия

<i>p</i>	% ИПС
0	100,00000000
1	50,00000000
2	25,00000000
3	12,50000000
4	6,25000000
5	3,12500000
6	1,56250000
7	0,78125000
8	0,39062500
9	0,19531250
10	0,09765625

Для обеспечения гибкости в использовании различных ГПСЧ для подключения этого модуля был разработан специальный интерфейс, состоящий из следующих функций: keygenopen, keyenclose, keygenreadNbits, keygenreadMask и структуры STEG_KEY. Рассмотрим назначение функций и структур подробнее.

Структура STEG_KEY является универсальным представлением ключевых данных произвольного ГПСЧ и имеет произвольную внутреннюю структуру, которая зависит от конкретного алгоритма ГПСЧ:

```
typedef char STEG_KEY;
```

Функция keygenopen предназначена для создания структуры STEG_KEY:

```
STEG_KEY *keygenopen(uint8_t thetext, uint8_t *pass, uint16_t passlen), где
```

thetext – начальное значение стеганографической ключевой последовательности;

pass – стеганографическая парольная фраза, задающая способ формирования стеганографической ключевой последовательности;

passlen – длина стеганографического ключа.

Функция keyenclose предназначена для завершения процесса создания стеганографической ключевой последовательности и удалении структуры bf из памяти:

```
int keyenclose(STEG_KEY *bf), где
```

bf – данные стеганографического ключа.

Функция `keygenreadNbits` предназначена для чтения очередного блока стеганографической ключевой последовательности:

`uint8_t keygenreadNbits (STEG_KEY *bf, uint8_t numOfBits)`, где

bf – данные стеганографического ключа;

numOfBits – количество бит, стеганографической ключевой последовательности, которые будут прочитаны (см. таблицу 2).

Функция `keygenreadMask` предназначена для создания шаблона, используемого в разработанном методе сокрытия:

`uint8_t keygenreadMask (STEG_KEY *bf, uint8_t numOfBits)`, где

bf – данные стеганографического ключа;

numOfBits – количество бит в созданном шаблоне (см. таблицу 2).

4.1.2. Реализация функциональных модулей, использующих открытые интерфейсы для подключения программного обеспечения стороннего разработчика

Для апробации разработанных в третьей главе открытых интерфейсов для подключения программного обеспечения стороннего разработчика были реализованы модули, осуществляющие чтение, запись и выделение пространства сокрытия из наиболее распространенных форматов хранения графических изображений и цифрового звука. Разработка модулей осуществлялась с использованием библиотек свободно распространяемых сторонними разработчиками.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате JPEG

Программное обеспечение, реализующее чтение и запись файлов, записанных в формате JPEG, разработано в виде динамически подключаемого модуля чтения и записи файлов JPEG. Он используется для чтения и записи файлов в формате JPEG, а также для осуществления доступа к пространству сокрытия.

Перечислим только основные функции данного модуля, которые необходимы для работы основного модуля:

`jpeg_std_error` – функция инициализации обработчика ошибок, необходима как для сжатия, так и для расжатия графических файлов в формате JPEG;

`jpeg_create decompress` – функция создания структуры, необходимой для осуществления расжатия файла JPEG;

`jpeg_stdio_src` – функция инициализации входного потока, содержащего файл JPEG;

`jpeg_read_header` – функция чтения заголовка файла JPEG и определения характеристик сжатого изображения, необходимых для его дальнейшего расжатия;

`jpeg_start_decompress` – функция начала процесса расжатия файла JPEG;

`jpeg_read_scanlines` – функция получения разжатых данных;

`jpeg_finish_decompress` – функция завершения процесса расжатия изображения;

`jpeg_destroy_decompress` – функция удаления структуры, необходимой для осуществления расжатия файла JPEG;

`jpeg_createcompress` – функция создания структуры, необходимой для осуществления сжатия файла JPEG;

`jpeg_stdio_dest` – функция инициализации выходного потока, в который будет записан файл JPEG;

`jpeg_set_defaults` – функция установки параметров и характеристик сжимаемого изображения;

`jpeg_set_quality` – функция установки качества (степени сжатия) сжимаемого изображения;

`jpeg_start_compress` – функция начала процесса сжатия файла JPEG;

`jpeg_write_scanlines` – функция записи разжатых данных;

`jpeg_finish_compress` – функция завершения процесса сжатия изображения;

`jpeg_destroy_compress` – функция удаления структуры, необходимой для осуществления сжатия файла JPEG.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате Wav, с поддержкой кодеков: PCM, A-Law, Mu-Law, Microsoft ADPCM, DVIM/IMA ADPCM, CCITT G.711, CCITT G.721, CCITT G.722

WAVE файлы являются подмножеством файлов RIFF формата (Resource Interchange File Format), разработанного для хранения ресурсов мультимедиа фирмой Microsoft. Как и в формате AIFF, основным элементом RIFF файла являются блоки (chunk). Каждый блок содержит свой 4-х байтовый идентификатор, размер данных и данные.

Идентификаторы строятся на основе последовательности 4-х символов, поэтому используют не числовые значения, а их символьное представление. Основные типы блоков имеют идентификаторы "RIFF" и "LIST" и могут состоять из вложенных блоков (подблоков).

Рассмотрим необходимый для реализации данного подключаемого модуля случай Wav файла, состоящего из одного лишь RIFF блока, содержащего WAVE блок. WAVE блок состоит из двух подблоков. Первый подблок имеет тип "fmt" и описывает формат следующих аудиоданных, в том числе он содержит следующую информацию:

- тип кодирования, которым закодированы последующие аудиоданные;
- число каналов (1 - моно, 2-стерео);
- частота дискретизации (число отсчетов в секунду);
- скорость байтового потока (среднее число байт в секунду, используется для эффективной буферизации);
- выравнивание данных в последующем блоке data.

Дополнительные данные, используемые для настройки соответствующего алгоритма сжатия/распаковки звукового потока.

Второй подблок имеет тип "data" и содержит непосредственно аудиоданные. Например, если тип кодирования – WAVE_FORMAT_PCM, то в этом блоке расположены непосредственно отсчеты звукового сигнала. Для моно сигнала отсчеты расположены непосредственно один за другим. Для стерео сигнала отсчеты чередуются по следующей схеме: 1-й отсчет левого канала, 1-й отсчет правого канала, 2-й отсчет левого канала, 2-й отсчет правого канала, и так далее.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате Aiff

Формат AIFF был предложен фирмой Apple и основан, в свою очередь, на формате IFF, предложенном фирмой Electronic Arts. Как и формат Microsoft Wave, данный формат представляет собой формат, основанный на вложенных блоках (chunk). Поскольку формат в первую очередь предназначался для использования на компьютерах фирмы Apple, то все числа в нем хранятся в формате MSF (Старшие байты вначале).

Каждый блок имеет следующую структуру:

```
struct {  
    ID      ckID;
```

```
    Int32_t ckDataSize;
    Char     ckData ;
};

здесь:
```

- ckID – идентификатор блока – 4-х байтовое число, составленное ASCII кодов 4-х символов;
- ckDataSize – размер блока данных, содержащихся в ckData;
- ckData – данные блока, выровненные до границы 2 байта, выравнивающий байт, если он присутствует, не учитывается в ckDataSize.

AIFF файлы состоят из одного глобального блока, имеющего идентификатор 'FORM'. Данный блок содержит структуру следующего вида:

```
struct {
    ID      formType;
    Char    Data ;
};

здесь:
```

- formType – идентификатор типа содержащихся данных, может принимать значения 'AIFF' или 'AIFC';
- Data – данные, содержащие вложенные блоки, описывающие формат звуковых данных, сами данные и другую информацию.

Файлы, содержащие форму типа 'AIFF' не могут содержать сжатые данные и как следствие практически не используются на практике. Отличие их от файлов с формой типа AIFC в основном состоит в отсутствии блока 'FVER'. В блоке Data описателя формата, обязательно содержатся блоки 'COMM' и 'SSND'.

Блок 'COMM' (Common chunk) используется для описания формата звуковых данных. Он имеет следующую структуру:

```
struct {
    ID      ckID;
    int32_t ckDataSize;
    int16_t numChannels;
    uint32_t numSampleFrames; /* # sample frames = samples/channel */
    int16_t sampleSize; /* # bits/sample */
    extended sampleRate; /* sample_frames/sec */
    ID      compressionType; /* compression type ID code */
    pstring compressionName; /* human-readable compression type name */
```

```
};
```

здесь:

- numChannels – количество звуковых каналов;
- numSampleFrames – количество кадров отсчетов (= количество отсчетов / количество каналов);

- sampleSize – число бит на отсчет;

- sampleRate – количество кадров отсчетов, воспроизводимых в секунду;

- compressionType – идентификатор типа компрессии;

- compressionName – описание типа компрессии.

Поля ckID и ckDataSize были рассмотрены ранее.

Блок 'SSND' используется для хранения непосредственно звуковых данных. Он имеет следующую структуру:

```
struct {  
    ID          ckID;  
    int32_t     ckDataSize;  
    uint32_t    offset;  
    uint32_t    blockSize;  
    char        soundData ;  
};
```

здесь:

- offset – смещение первого отсчета от начала звуковых данных в байтах;

- blockSize – используется совместно с полем offset для задания длины звуковых

данных;

- soundData – непосредственно закодированные, заданным в блоке 'COMM' кодеком, данные.

Поля ckID и ckDataSize были рассмотрены ранее.

Блок 'FVER' (Format version) используется для указания версии формата AIFF, использованного при создании файла. Он имеет следующую структуру:

```
struct {  
    ID          ckID;  
    int32_t     ckDataSize;  
    uint32_t    timestamp;  
};
```

Поля здесь ckID и ckDataSize были описаны ранее. Поле timestamp представляет собой время создания спецификации формата файла. На данный момент поле timestamp всегда равно значению 0xA2805140.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате Snd

Файловый формат Snd был предложен фирмой Sun и впоследствии использован фирмой NeXT. Первоначально он представлял собой просто копию памяти, содержащую звуковые данные и относился к форматам без заголовка. В дальнейшем в формат был добавлен заголовок, позволяющий производить идентификацию этих потоков и описывать свойства аудиосигнала.

Основным достоинством формата Snd является его простота. Заголовок определяет все необходимые свойства аудиосигнала – такие как формат кодирования, частота дискретизации, число каналов. Звуковые данные хранятся после заголовка в виде одного непрерывного блока, который может представлять собой просто копию памяти.

Структура, описывающая заголовок файла в формате Snd определяется следующим образом:

```
struct {  
    int magic;  
    int dataLocation;  
    int dataSize;  
    int dataFormat;  
    int samplingRate;  
    int channelCount;  
    char info[4];  
};
```

Рассмотрим составляющие ее поля более подробно.

Поле magic содержит идентификатор ("магическое число") файла. На основе данного поля можно произвести идентификацию файла как файла в формате Snd. Идентификатор содержит 4-х байтовое число, состоящее из ASCII кодов строки ".snd", то есть он равен 0x2E736E64.

Поле dataLocation содержит четырехбайтовое целое число, содержащее положение первого байта звуковых данных от начала файла. Необходимость в данном поле

обусловлена тем, что заголовок содержит поля переменного размера. Тем не менее, на практике, данное число чаще всего равно 28.

Поле dataSize содержит четырехбайтовое целое число, содержащее размер звуковых данных в байтах. При потоковом сохранении файлов в формате Snd не всегда с самого начала известно количество сохраняемых данных. Следовательно, не всегда возможно корректно сформировать заголовок, в частности поле dataSize. Для решения этой проблемы в поле dataSize записывают некоторое большое число, заведомо большее размера данных, которые будут записаны в файл. Таким образом, при чтении файла необходимо проверять условия конца файла, поскольку нет гарантии того, что в файле содержится именно dataSize байтов аудиоинформации.

Поле dataFormat содержит четырехбайтовое целое число, описывающее формат последующих звуковых данных. Данное поле может принимать одно из следующих значений:

- 1 – 8-битные отсчеты в кодировке MuLaw;
- 2 - 8-битные отсчеты;
- 3 - 16-битные отсчеты;
- 4 - 24-битные отсчеты;
- 5- 32-битные отсчеты;
- 6 – отсчеты в формате с плавающей точкой;
- 7 – отсчеты в формате с плавающей точкой двойной точности;
- 8 – фрагментированные отсчеты;
- 10 – в блоке данных содержится код программы;
- 11 – 8-битные отсчеты с фиксированной точкой;
- 12 – 16-битные отсчеты с фиксированной точкой;
- 13 – 24-битные отсчеты с фиксированной точкой;
- 14 – 32-битные отсчеты с фиксированной точкой;
- 23 – АДИКМ кодированные отсчеты с использованием кодека G721 со скоростью битового потока 32 кбит/с и 4 битами на отсчет;
- 24 – АДИКМ кодированные отсчеты с использованием кодека G722
- 25 – АДИКМ кодированные отсчеты с использованием кодека G723 со скоростью битового потока 24 кбит/с и 3 битами на отсчет;

- 26 – АДИКМ кодированные отсчеты с использованием кодека G723 со скоростью битового потока 40 кбит/с и 5 битами на отсчет;
- 27 – 8-битные отсчеты в кодировке ALaw.

Поле samplingRate содержит четырехбайтовое целое, задающее частоту отсчетов в секунду.

Поле channelCount содержит число звуковых каналов, содержащихся в аудиоинформации. Соответственно один канал означает моно сигнал, два канала стерео сигнал и так далее.

Поле info содержит текстовую строку, описывающую содержащиеся в файле звуковые данные, заканчивающуюся нулем. Размер данного поля является переменной величиной и определяется на основе поля dataLocation.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате Vos

Формат Vos был впервые предложен фирмой Creative Labs для использования в своих продуктах. В файлах Vos могут содержаться только моно или стерео звуковые данные, закодированные в виде 8 или 16 битных отсчетов или в форматах Alaw, MuLaw.

Файл, содержащий данный формат начинается с заголовка следующего вида:

```
struct {
    char txt_id[20];
    unsigned int16_t offset_data;
    unsigned int16_t ver;
    unsigned int16_t id;
};
```

здесь

- txt_id – текстовый идентификатор, служащий для распознавания файлов в формате VOC он всегда равен "Creative Voice File\x1A".

- offset_data – число, указывающее смещение начала первого блока данных.

- ver – номер версии формата.

i- d - идентификатор, число формируемое на основе номеру версии плюс по следующему правилу: id = (((~ver)+0x1234) & 0xFFFF)

Затем идут блоки данных. Они разбиты на подблоки разных типов. Структура подблока данных:

```
struct {  
    unsigned char type;  
    unsigned int24_t len;  
};
```

здесь

- type - тип подблока;
- len - длина подблока.

Далее идут данные подблока. В зависимости от типа подблока они разные.

Возможны следующие типы подблоков:

type=0 - Завершает блок данных.

type=1 - Подблок звуковых данных. При этом из длины подблока нужно вычесть два, чтобы получить длину звуковых данных. Этот подблок имеет свой добавочный заголовок следующего вида:

```
struct {  
    unsigned char rate;  
    unsigned char type_zip;  
};
```

здесь:

- rate - частота дискретизации, для получения этого значения в Гц необходимо использовать следующую формулу: Частота_Гц = 1000000 / (256 – rate);

- type_zip - способ упаковки: всегда равен 0 - не упакованы.

Далее идут звуковые данные в виде без знаковых 8 битных целых чисел.

type=2 - Подблок продолжения звуковых данных.

type=3 - Подблок тишины. Содержит структуру следующего вида:

```
struct {  
    unsigned int16_t time;  
    unsigned char rate;  
};
```

здесь:

- time - время тишины, для получения этого значения в Гц необходимо использовать следующую формулу: Частота_Гц = 1000000 / (256 – rate);

- rate - частота дискретизации.

type=4 - Подблок-маркер, содержит без знаковое 16 битное целое.

type=5 - Подблок текстовых сообщений. Текстовые данные, заканчивающиеся нулем.

type=6 - Подблок начала звукового цикла. Содержит беззнаковое 16 битное целое, равное числу повторов минус единица.

type=7 - Подблок конца звукового цикла. Не содержит данных

type=8 - Подблок расширенных данных. Содержит структуру следующего вида:

```
struct {  
    unsigned int16_t rate;  
    unsigned char pack;  
    unsigned char mode;  
};
```

здесь:

- **rate** - частота дискретизации, для получения этого значения в Гц необходимо использовать следующие формулы:

- для моно сигнала: Частота_Гц = 256000000 / (65536 - Частота_B_Файле);
- для стерео сигнала: Частота_Гц = 128000000 / (65536 - Частота_B_Файле).
- **pack** - способ упаковки, всегда равно 0 - без упаковки;
- **mode** - Режим : 0 - моно, 1 – стерео.

Далее идут звуковые данные в виде без знаковых 8 битных целых чисел.

type=9 - Подблок расширенных данных. Содержит структуру следующего вида:

```
struct {  
    unsigned int32_t rate;  
    unsigned char bitwidth;  
    unsigned char mode;  
    unsigned int16_t encoding;  
    unsigned int32_t reserved;  
};
```

здесь:

- **rate** - частота дискретизации, для получения этого значения в Гц необходимо использовать следующие формулы:

- для моно сигнала: Частота_Гц = 256000000 / (65536 - Частота_B_Файле);
- для стерео сигнала: Частота_Гц = 128000000 / (65536 - Частота_B_Файле).
- **bitwidth** – количество бит на отсчет;

- encoding - способ упаковки отсчетов: 0 – без знаковое 8 битное целое, 4 – 16 битное целое, 6- ALaw, 7 - MuLaw;
 - mode - Режим : 0 - моно, 1 – стерео.
- Далее идут звуковые данные.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных в формате Pcm, Sam, Raw

Форматы PCM, Sam, Raw относятся к форматам без заголовка, и содержат только выборки звукового сигнала. Выборки могут иметь произвольную частоту следования в выборках в секунду, содержать один, два или большее число звуковых каналов. Тесм самым для обработки этих файлов, необходимо обладать следующей дополнительной информацией:

- число бит на выборку (8/16/24/32 бита);
- частота выборки (от 5 до 48кГц);
- число каналов (1/2-моно/стерео, возможно до 4 каналов).

Файлы в формате Au являются файлами в формате Snd. Различие в наименовании обусловлено историческими причинами. Файлы в формате Au изначально использовались на платформе Sun. Затем формат был перенесен на платформу NeXT и переименован в в формат Snd.

Разработка программного обеспечения, реализующего чтение и запись файлов, записанных с использованием расширения ACM (Audio Compression Manager) Microsoft Windows.

Подсистема сжатия звука (Audio Compression Manager — ACM, диспетчер сжатия звука) первоначально была введена в расширении Video For Windows (VFW) для Windows 3.x, вместе с подсистемой сжатия изображения (Video Compression Manager — VCM). Целью введения этих подсистем было создание удобного, универсального и гибкого интерфейса для работы со звуковыми и видеопотоками. Позднее ACM и VCM были включены в состав стандартной звуковой подсистемы для платформ Win32. Таким образом, для работы с ACM необходима либо платформа Windows 3.x с расширением VFW, либо платформа Win32.

Подсистема сжатия звука предоставляет приложениям набор сервисных средств для преобразования звуковых форматов и других видов обработки звуковых данных. Она реализована в виде самого диспетчера ACM и набора так называемых драйверов ACM — независимо подключаемых программ, обеспечивающих собственно обработку звука. Приложения непосредственно взаимодействуют только с диспетчером, который выбирает нужные драйверы, передает информацию между приложением и драйверами и выполняет прочие функции по координации работы подсистемы.

В ACM существует два вида обработки звука: преобразование формата — изменение способа представления звуковых данных, перекодировка их из одного представления в другие без изменения общих свойств самого звука; фильтрование — обработка звука в потоке (усиление/ослабление, изменение АЧХ, наложение звуковых эффектов и т.п.) без изменения формата.

Модули драйверов ACM, реализующие функции преобразования форматов, называют преобразователями формата (format convertors). Модули, реализующие функции фильтрования, называются фильтрами (filters). Один и тот же подключаемый драйвер может совмещать в себе различные функции, а также содержать более одного преобразователя или фильтра.

Преобразование форматов, при котором меняется тип (tag) формата, обычно сопровождается изменением объема звуковых данных. Такие виды преобразования называются сжатием (compression) или восстановлением (decompression). Сжатию обычно подвергаются данные в формате PCM, а при восстановлении формат PCM обычно имеет результат операции. Преобразователи формата, реализующие сжатие/восстановление, называются кодеками (codec — coder/decoder или compressor/decompressor).

Преобразование данных в ACM производится порциями. Порция потока преобразования обычно имеет размер от нескольких десятков байт до сотен килобайт, в зависимости от структуры звукового формата и динамики преобразования. За каждую операцию преобразования обрабатывается одна порция данных.

Для каждого потока преобразования создаются два буфера: входной — для порции исходных данных, и выходной — для порции данных результата. Размер преобразуемой порции не может превышать выбранного размера буфера.

ACM обрабатывает звуковые данные потоками. Это означает, что приложение не может обрабатывать звуковые данные в произвольной последовательности, а делает это

только последовательно, от начала к концу потока. Такое ограничение связано с тем, что как преобразование форматов, так и фильтрование может потребовать информации о предшествовании — например, относительное кодирование амплитуд в формате ADPCM или формирование эффекта эха в фильтре Echo. Поэтому обработка звукового потока в ACM включает в себя три стадии:

- создание, или открытие потока (acmStreamOpen). При этом приложение сообщает ACM параметры потока (форматы, режимы обработки, способы уведомления) и получает от ACM ключ открытого потока.

- обработка потока (acmStreamConvert). На этом этапе приложение последовательно передает ACM буферы обмена, содержащие порции преобразуемых звуковых данных. ACM передает буферы выбранному драйверу для обработки, а после завершения обработки каждый буфер возвращается приложению. Обработка потока может происходить в реальном времени, если аппаратура компьютера успевает выполнять ее достаточно быстро.

Уничтожение, или закрытие потока (acmStreamClose). Последняя операция с потоком, после которой поток перестает существовать.

Поток преобразования ACM фактически состоит из двух звуковых потоков — исходного (source) и результирующего (destination). Данные исходного потока существуют до начала обработки и не изменяются после ее завершения; данные результирующего потока создаются в процессе обработки.

Запрос сведений о заданном фильтре или формате

Для имеющегося описателя фильтра или описателя формата выполняются функции acmFilterTagDetails, acmFilterDetails или acmFormatTagDetails, acmFormatDetails, возвращающие сведения либо о типе, к которому принадлежит указанный фильтр/формат, либо о самом фильтре/формате. В частности, полученные сведения могут использоваться для вывода информации пользователю.

Перебор доступных типов и стандартных фильтров/форматов

Для имеющегося набора условий, которым должен удовлетворять искомый тип или стандартный фильтр/формат, выполняются функции перебора фильтров/форматов — acmFilterTagEnum, acmFilterEnum, acmFormatTagEnum, acmFormatEnum. В результате для каждого подходящего типа или для каждого подходящего стандартного фильтра/формата вызывается соответствующая функция поддержки перебора. В результате поиск может

быть прерван при обнаружении подходящего объекта или может быть составлен список объектов, который затем предлагается пользователю для ручного выбора.

Выбор фильтра/формата при помощи стандартного диалога

Для имеющегося набора условий, которым должен удовлетворять искомый тип или стандартный фильтр/формат, формируется структура ACMFILTERCHOOSE / ACMFORMATCHOOSE и выполняется функция выбора acmFilterChoose / acmFormatChoose. В результате ACM выводит пользователю стандартный диалог, содержащий списки подходящих типов и стандартных фильтров/форматов. После выбора пользователем нужного фильтра/формата ACM возвращает его описание и дополнительные сведения.

Выбор наиболее подходящего формата

В том случае, когда имеется сжатый формат, который должен быть восстановлен в формат PCM, приложение может воспользоваться функцией acmFormatSuggest. Эта функция подберет наиболее подходящий формат типа PCM, в который следует восстановить исходный сжатый формат.

Обработка потока данных

Для выполнения необходимой обработки исходного потока данных приложение создает поток преобразования при помощи функции acmStreamOpen. При создании потока преобразования указываются исходный и результирующий форматы либо фильтр, которым должен быть обработан исходный поток. Затем приложениис выполняет серию операций преобразования, вызывая функцию acmStreamConvert, каждый вызов которой преобразует очередную порцию исходных данных, порождая в результате порцию результирующих данных.

Исходный звуковой поток может быть взят из файла или получен в реальном времени — от любого звукового устройства, по сети или другим путем. Результирующий поток также может быть записан в файл, выведен в реальном времени на звуковое устройство, передан по сети и т.п.

Функции ACM. Классы функций интерфейса ACM

Интерфейс ACM имеет в своем составе четыре основных класса интерфейсных функций. Принадлежность функции к определенному классу обозначается префиксом в ее имени:

- функции работы с драйверами (acmDriver). Служат для установки, удаления, настройки, опроса и перебора доступных драйверов ACM.
- функции работы с фильтрами (acmFilter). Служат для опроса, перебора и выбора доступных фильтров.
- функции работы с форматами (acmFormat). Служат для опроса, перебора и выбора доступных форматов (кодеков).
- функции работы с потоками (acmStream). Служат для создания потоков преобразования и обработки порций данных в них.

Информационные функции

acmGetVersion	- Запрос версии ACM
acmMetrics	- Запрос различных параметров ACM

Функции работы с драйверами

acmDriverAdd	Установка собственного драйвера приложения или задание окна уведомления
acmDriverRemove	- Удаление собственного драйвера приложения
acmDriverOpen	- Открывание драйвера
acmDriverClose	- Закрывание драйвера
acmDriverDetails	- Запрос сведений о драйвере
acmDriverEnum	- Перебор доступных драйверов
acmDriverID	- Запрос идентификатора драйвера
acmDriverPriority	- Установка приоритета драйвера
acmDriverMessage	- Передача сообщения драйверу

Функции работы с фильтрами

acmFilterTagDetails	- Запрос сведений о типе фильтра
acmFilterDetails	- Запрос сведений о фильтре
acmFilterTagEnum	- Перебор доступных типов фильтров
acmFilterEnum	- Перебор доступных стандартных фильтров
acmFilterChoose	- Выбор фильтра при помощи стандартного диалога ACM

Функции работы с форматами

acmFormatTagDetails	- Запрос сведений о типе формата
acmFormatDetails	- Запрос сведений о формате

acmFormatTagEnum	- Перебор доступных типов форматов
acmFormatEnum	- Перебор доступных стандартных форматов
acmFormatChoose	- Выбор формата при помощи стандартного диалога ACM
acmFormatSuggest	- Запрос наиболее подходящего для преобразования формата

Функции работы с потоками

acmStreamOpen	- Открывание потока преобразования
acmStreamClose	- Закрывание потока
acmStreamSize	- Запрос размеров буферов потока
acmStreamPrepareHeader	- Подготовка буфера потока
acmStreamUnprepareHeader	- Отмена подготовки буфера потока
acmStreamConvert	- Преобразование очередной порции данных в потоке
acmStreamReset	- Сброс (уничтожение) потока
acmStreamMessage	- Передача сообщения драйверу потока

4.1.3. Реализация интерфейсов для взаимодействия с пользователем

Помимо программных интерфейсов, предназначенных для подключения дополнительных модулей, главный программный модуль агента управления имеет пользовательский интерфейс, позволяющий наглядно демонстрировать возможности разработанного комплекса. На главной форме пользовательского интерфейса управления агентом регистрации мультимедиа информации (см. рисунок 35) расположена кнопка для вызова диалога для открытия мультимедиа-контейнера, поле для ввода стеганографической паролевой фразы, кнопки "Извлечь ЦВЗ", "Сокрыть ЦВЗ", "Емкость контейнера", регуляторы плотности сокрытия и качества модифицированного контейнера (в случае если его алгоритм поддерживает сжатие с потерями).

В случае, когда возникают ошибки при загрузке обязательных программных модулей, несущих в себе функциональность необходимую для работы агентов комплекса, пользователю выдается сообщение об ошибке (см. рисунок 37) и работа соответствующих агентов комплекса прекращается.

Во время работы с контейнером (см. рисунок 36) становятся доступными функциональные кнопки, в окне справа отображается информация и мультимедиа-контейнер, а внизу информация о его характеристиках и о характеристиках пространства

сокрытия, полученных с учетом введенной парольной фразы, выбранной плотности сокрытия и качества модифицированного контейнера.

Также пользователю доступна информация о подключенных при загрузке комплекса модулях (см. рисунок 38).

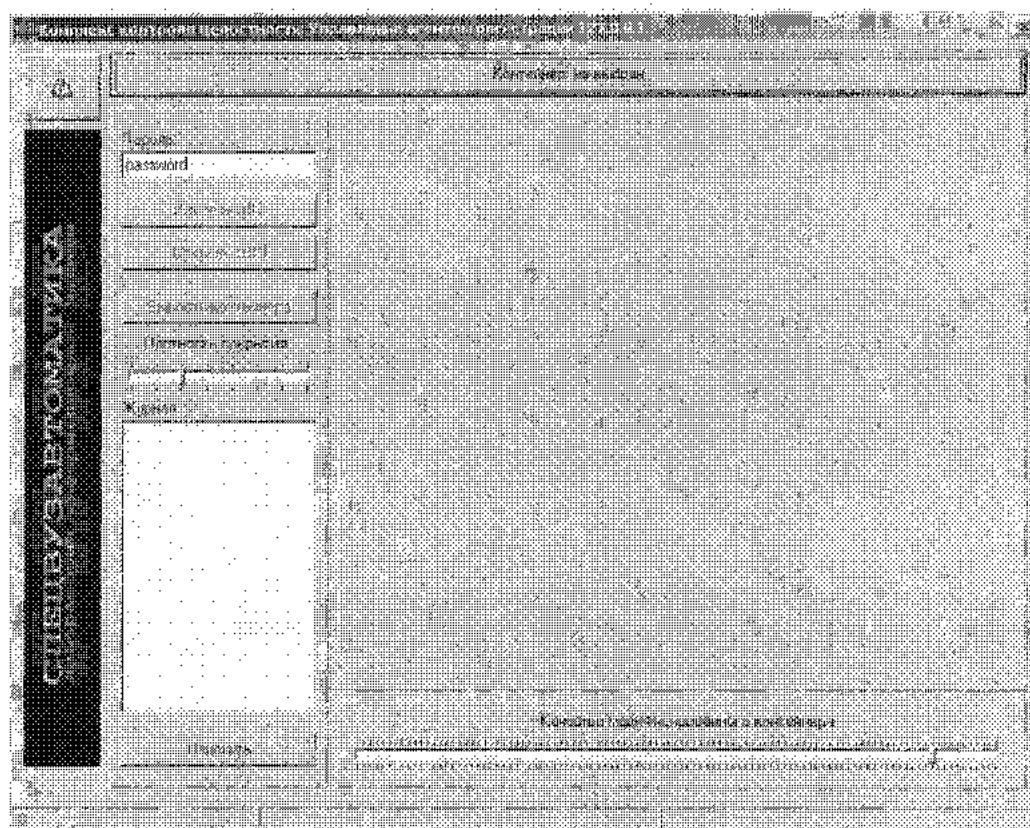


Рисунок 35 – Форма управления агентом регистрации мультимедиа информации

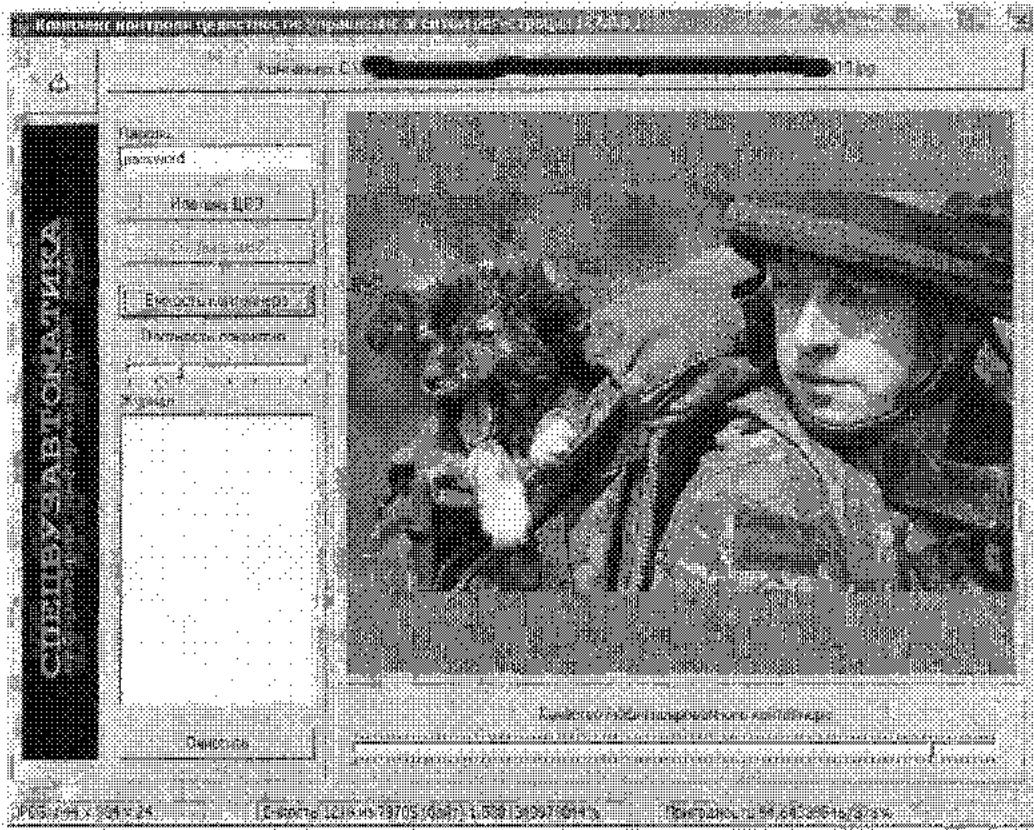


Рисунок 36 – Форма управления агентом регистрации мультимедиа-информации во время работы с пустым контейнером

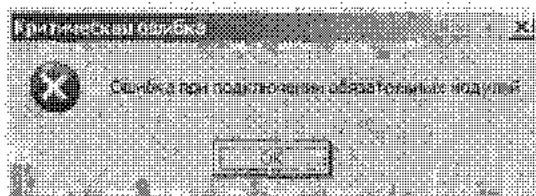


Рисунок 37 – Сообщение об ошибке, возникшей при загрузке обязательных модулей

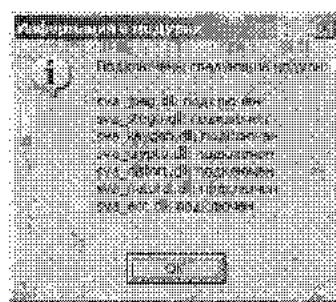


Рисунок 38 – Форма с информацией о подключенных модулях

4.2. Экспериментальная оценка аспектов функционирования комплекса, вносимых искажений и нарушения естественности мультимедиа-контейнеров

Для оценки скоростных характеристик программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков был проведен соответствующий эксперимент. Для его проведения использовался набор изображений в формате JPEG полученных при помощи цифровой фотокамеры Nikon D70. Общий объем используемых в teste данных составил 900 Мегабайт (около 300 изображений). Эти изображения обрабатывались агентами регистрации мультимедиа-информации, в них внедрялся ЦВЗ, содержащий код контроля целостности, затем они добавлялись в информационное хранилище комплекса.

Для обработки использовались компьютеры Pentium IV с частотой 2,4 ГГц, оперативной памятью 512 мегабайт с операционной системой Microsoft Windows XP. Компьютеры были объединены в локальную сеть при помощи коммутатора D-Link 100 Мбит/с.

На рисунке 39 представлен график, отображающий средний поток изображений, обрабатываемый комплексом в различных конфигурациях. По оси абсцисс отложено количество компьютеров с установленными агентами регистрации мультимедиа-информации, используемые одновременно для обработки изображений, по оси ординат – поток данных в килобайтах в секунду (Кб/сек.). Кроме того, показаны скорости работы при различных конфигурациях комплекса:

1. на компьютере с агентом управления была установлена СУБД и работающий агент регистрации мультимедиа-информации;
2. на компьютере с агентом управления была установлена СУБД;
3. все агенты и СУБД установлены на разных компьютерах.

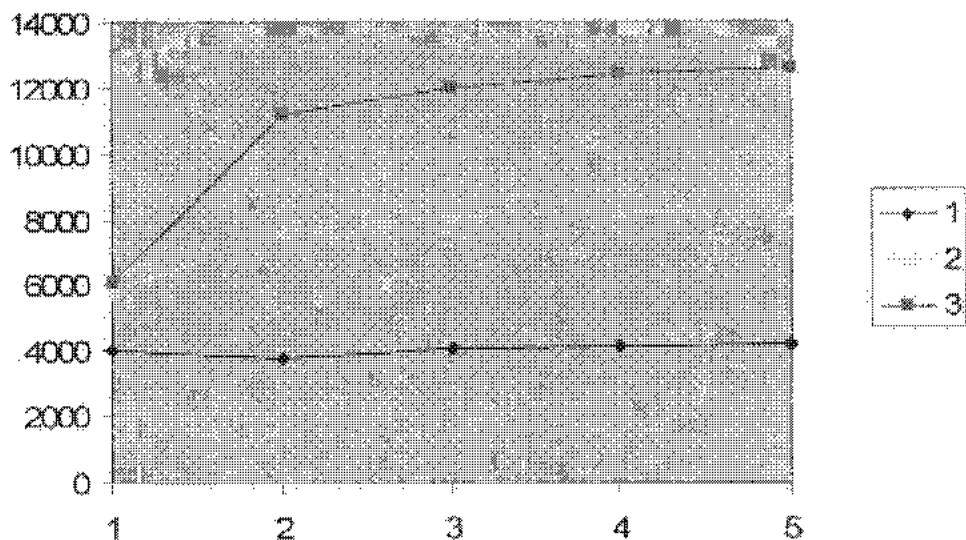


Рисунок 39 – Скоростные характеристики программного комплекса информационной защиты мультимедиа-информации

Для проведения оценки стойкости разработанных методов сокрытия данных в соответствии с разработанной моделью естественности был сформирован тестовый набор из 1000 изображений в формате JPEG и 1000 аудио-файлов в формате Wav, который состоял из:

- 250 изображений взятых со страниц сети Интернет;
- 500 изображений полученных при помощи цифровых фотоаппаратов (Nikon D70, Nikon CoolPix E995, Canon 300D, и др.);
- 250 изображений полученных при помощи сканирования (Agfa E52, Nikon CoolScan 4000ED, Epson 2400PHOTO).
- 900 файлов Wav содержащих музыкальные композиции;
- 100 файлов Wav содержащих записи дикторов.

Для оценки возможности автоматического принятия решения о наличии в мультимедиа-контейнере сокрытой информации использовалось решающее правило, основанное на критерии Неймана-Пирсона. При составлении решающего правила, было сделано предположение о нормальном распределении значений характеристик для пустого и заполненного контейнера. Использование этого предположения и полученных значений различных характеристик, позволило получить значения ошибок первого и второго рода для различных характеристик и различных степеней сокрытия.

Эксперимент проводился в соответствии с разработанной моделью естественности (см. Главу 2) [75]:

1. В тестовый набор были скрыты данные с различными плотностями скрытия;
2. Для всех мультимедиа-контейнеров были вычислены параметры естественности;
3. Для каждого параметра в отдельности был использован критерий Неймана-Пирсона и получены ошибки первого и второго рода;
4. Среди всех полученных ошибок первого и второго рода были взяты минимальные значения.

Часть полученных результатов представлена в таблице 3. Из которой видно, что при степени скрытия больше 4 не происходит нарушение естественности мультимедиа-информации 82.

Таблица 3 – Зависимость нарушения естественности от плотности скрытия

Степень скрытия	1	2	3	4	5	6
Ошибка первого рода	0,050957	0,296301	0,475512	0,5	0,5	0,5
Ошибка второго рода	0,118813	0,379463	0,421825	0,467799	0,489587	0,495317

Оценка вносимых при скрытии ЦВЗ искажений проводилась в соответствии с разработанной моделью и показала, что искажения, вносимые при скрытии со степенью скрытия больше 4 не значительны, т.с. не могут оказать влияние на перцептивное качество мультимедиа-информации. Наибольшие искажения возникают в тех случаях, когда размеры пространства невелики (при увеличении плотности скрытия). Наиболее наглядные результаты показаны на рисунках 40-53, где сплошной линией обозначены искажения, возникающие при сжатии JPEG, а прерывистой при сжатии JPEG с скрытием во все младшие биты (т.е. при максимальной плотности скрытия).

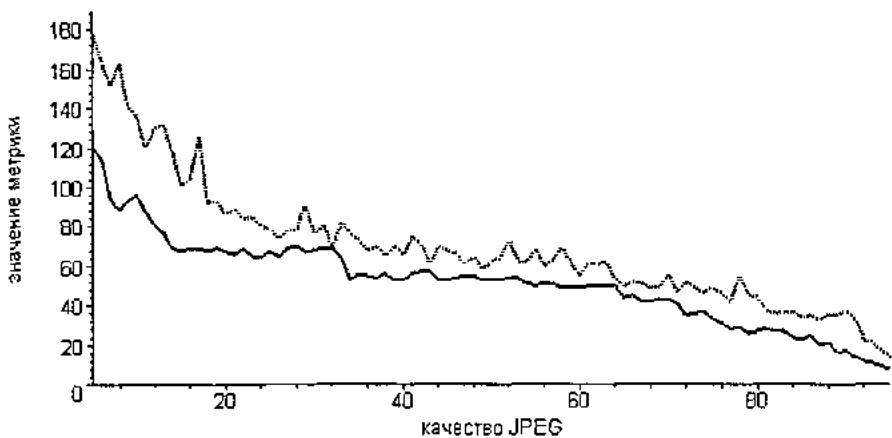


Рисунок 40 – Наибольшее отклонение

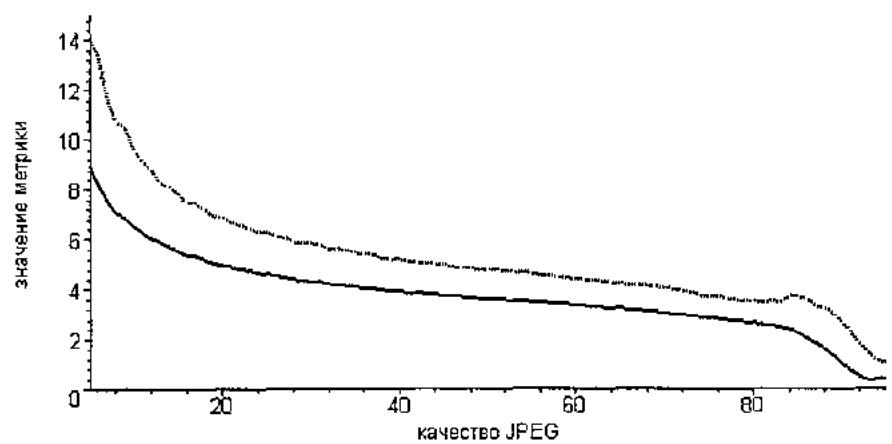


Рисунок 41 – Среднее по модулю отклонение

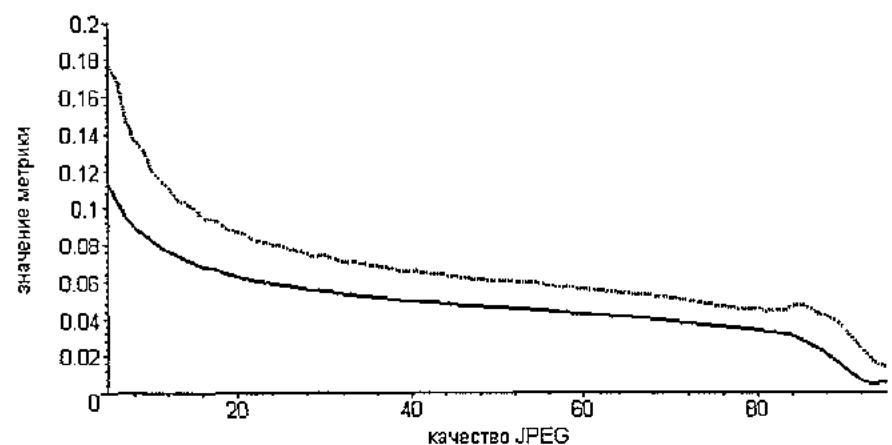


Рисунок 42 – Нормированное среднее по модулю отклонение

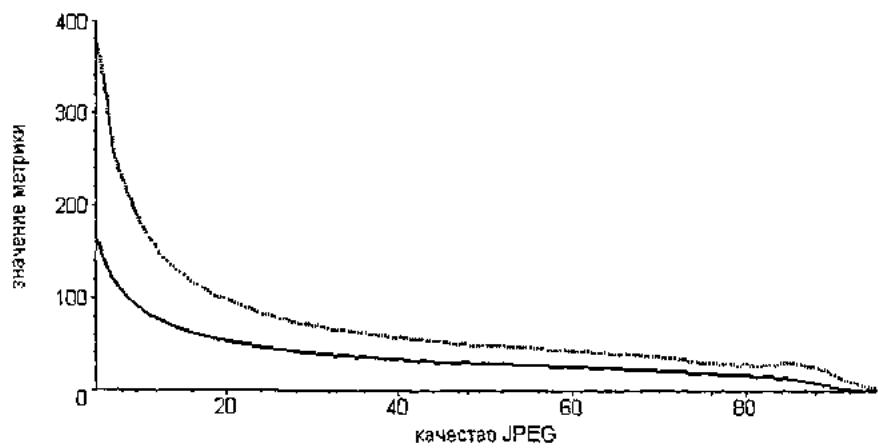


Рисунок 43 – Средне-квадратичное отклонение

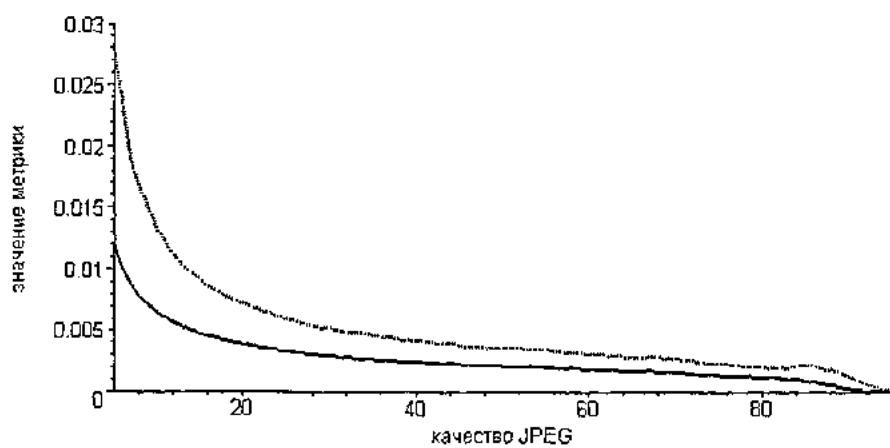


Рисунок 44 – Нормированное средне-квадратичное отклонение

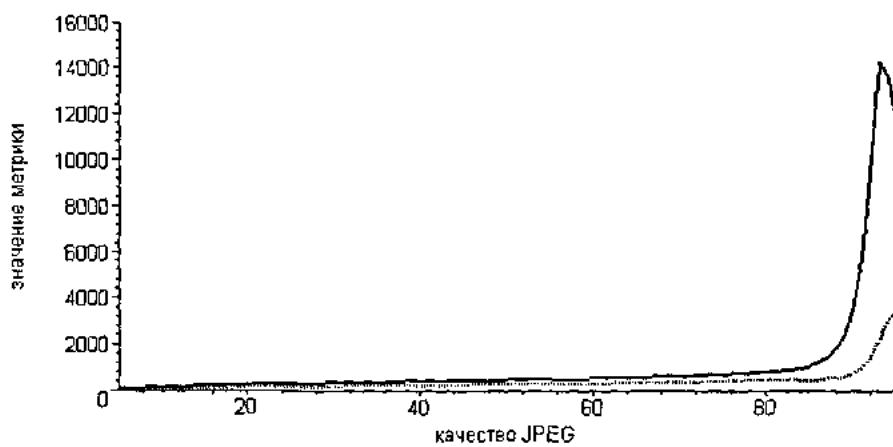


Рисунок 45 – Отношение сигнал-шум

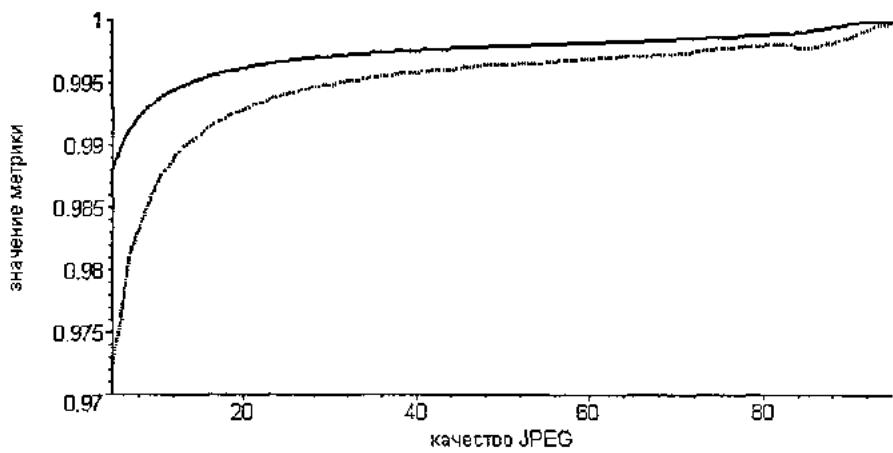


Рисунок 46 – Точность изображения

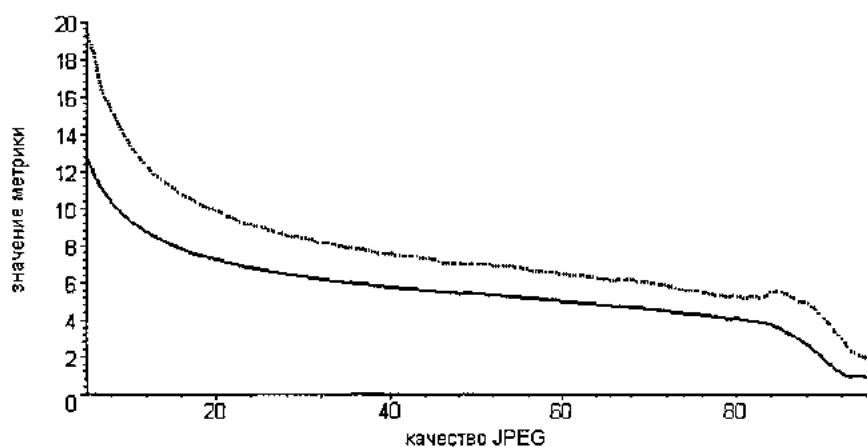


Рисунок 47 – L₄-норма

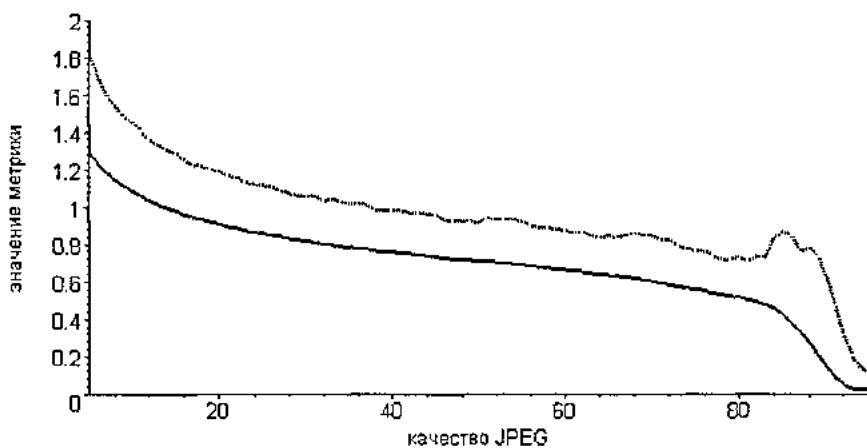


Рисунок 48 – Средне-квадратичное отклонение лапласиана

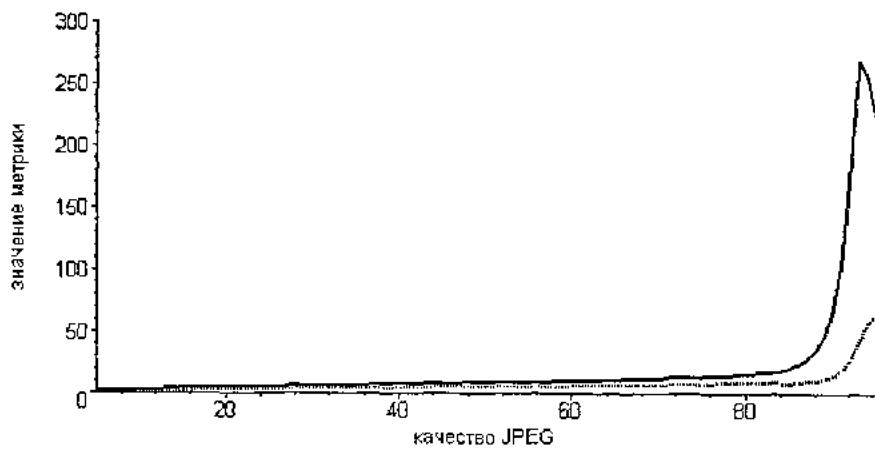


Рисунок 49 – Пиковое отношение сигнал-шум

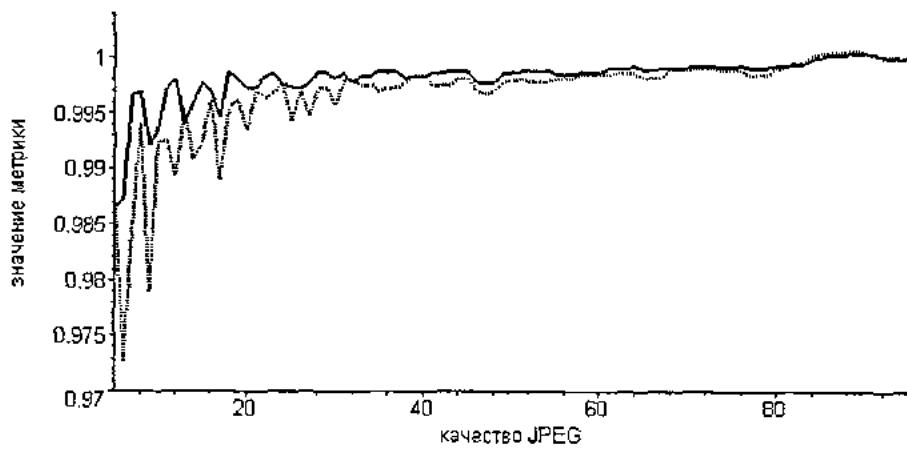


Рисунок 50 – Нормированная взаимная корреляция

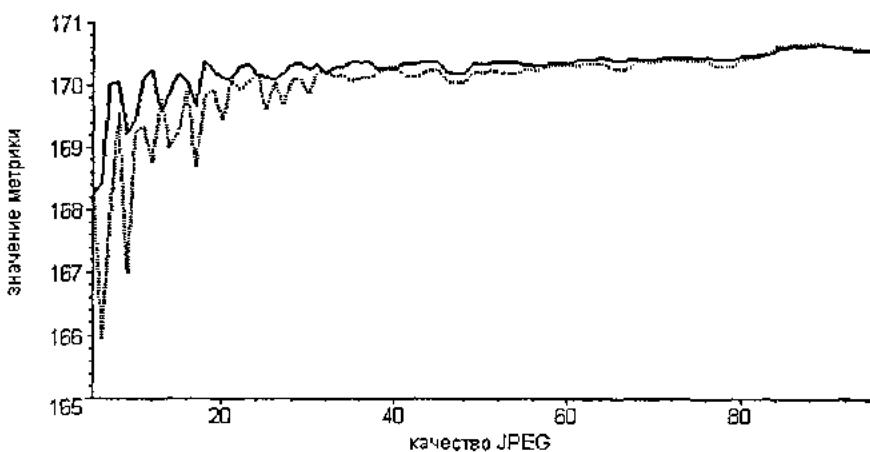


Рисунок 51 – Корреляция качества

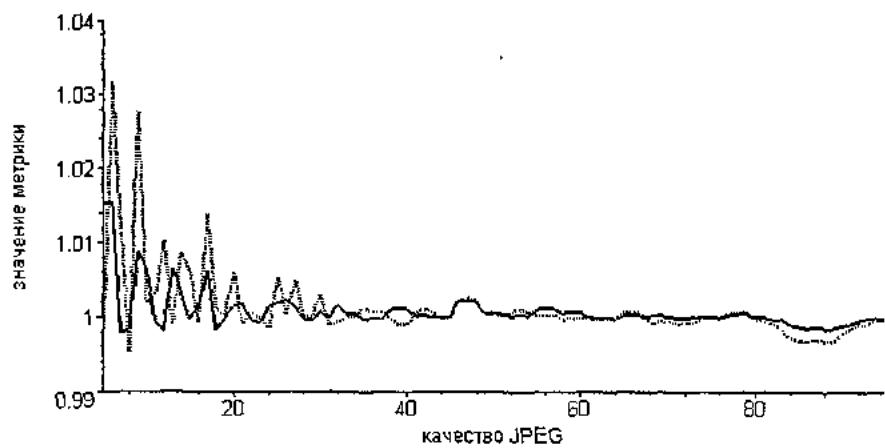


Рисунок 52 – Конструктивный контент

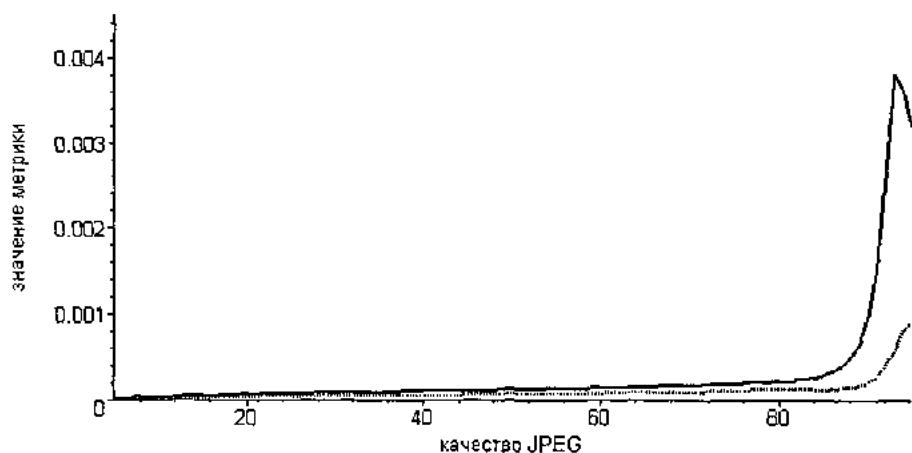


Рисунок 53 – Отношение сигма-сигнала к среднему квадратичному отклонению

4.3. Выводы

1. В данной главе описаны вопросы практической реализации разработанной архитектуры программного комплекса контроля целостности мультимедиа-информации с учетом разработанной модели естественности и методов внедрения и извлечения ЦВЗ. На основе разработанной архитектуры программного обеспечения, математических моделей и методов для демонстрации возможности реализации предлагаемых принципов была разработана программная система контроля целостности мультимедиа-информации.

2. Разработанные в данной главе программные средства предназначены для организации распределенного хранения мультимедиа-информации в информационно-телекоммуникационных системах и контроля ее целостности и построены с использованием функциональных модулей, использующих как универсальные интерфейсы взаимодействия управляющих и функциональных компонент, так и открытые интерфейсы для подключения программного обеспечения стороннего разработчика.

3. Проведена экспериментальная оценка аспектов функционирования комплекса, вносимых при сокрытии искажений и нарушению естественности мультимедиа-контейнеров, которая позволила определить максимально возможную плотность сокрытия, не приводящую к внесению статистических искажений. Анализ, проведенный при помощи разработанной модели оценки перцептивных искажений, показал что искажения, вносимые при сокрытии с допустимой с точки зрения нарушения естественности плотностью сокрытия, не нарушают перцептивное качество контейнера.

ЗАКЛЮЧЕНИЕ

Диссертация посвящена разработке программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров.

Для этого были проанализированы существующие угрозы безопасности мультимедиа-информации, рассмотрены первичные классы угроз. В результате анализа были выявлены основные угрозы безопасности мультимедиа-информации: нарушение конфиденциальности и нарушение целостности. Были исследованы средства и методы внедрения цифровых водяных знаков, рассмотрены их основные свойства. Рассмотрены основные типы атак на системы внедрения цифровых водяных знаков и методы оценки стойкости систем внедрения информации, необходимые для разработки методов внедрения и их последующего анализа. Проанализированы криптографические преобразования, которые являются основой систем защиты мультимедиа-информации и контроля ее целостности, а именно симметричные шифры, бесключевые хэш-функции, хэш-функции с зависимостью от секретного ключа.

В процессе проведения исследования были выявлены недостатки, присущие всем перечисленным методам и системам. А именно, существующие системы внедрения водяных знаков обладают низкой стойкостью к атакам активных и пассивных противников, при этом в них отсутствуют методы анализа нарушения перцептивного качества и статистических свойств мультимедиа-контейнеров, затрудняющие принятие решения о возможности их использования на практике. Существующие системы контроля целостности основаны на использовании хэш-функций, недостатком такого подхода является необходимость организации хранения вычисленных хэш-значений и установления связи между объектом контроля и хэш-значением.

С учетом выявленных недостатков и актуальности выбранной темы исследования была поставлена цель диссертационного работы, которая заключается в разработке программного комплекса информационной защиты мультимедиа-информации с использованием специальных цифровых водяных знаков, предназначенных для внедрения

идентификационных, классификационных данных и кода контроля целостности мультимедиа-контейнеров.

В соответствии с поставленной целью были сформулированы следующие научные задачи исследования:

- разработать модель естественности мультимедиа-информации, предназначенную для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков;

- разработать методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров;

- разоработать метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков;

- разработать модель оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки;

- разработать обобщенную архитектуру программного комплекса контроля целостности мультимедиа-информации, включающую интерфейсы взаимодействия управляющих и функциональных компонент и открытые интерфейсы для подключения программного обеспечения стороннего разработчика;

Была разработана модель естественности мультимедиа-информации, предназначенная для статистической оценки изменений мультимедиа-контейнера, вызванных внедрением цифровых водяных знаков. Разработанная модель может быть использована для различных типов и форматов хранения мультимедиа-информации за счет введенного в ней понятия пространства сокрытия. Использование данной модели не ограничивается оценкой стойкости к обнаружению сокрытой информации, ее вторым назначением является контроль естественности контейнеров перед внедрением в них ЦВЗ (определение пригодности контейнеров).

На основе модели естественности разработаны стойкие методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров. Данные методы зависят от секретного ключа, однозначно определяющего способ внедрения и извлечения информации и позволяют равномерно распределять внедряемую информацию по пространству сокрытия.

Также был разработан метод контроля целостности мультимедиа-контейнеров, основанный на внедрении специальных цифровых водяных знаков. Данный метод

позволяет использовать стандартные хэш-функций, но при этом для осуществления контроля целостности не требуется хранить хэш-значение.

Была разработана модель оценки вносимых при внедрении цифровых водяных знаков искажений и методы их количественной оценки. Данная модель необходима для получения разносторонних численных характеристик, позволяющих оценить ухудшения перцептивного качества контейнеров.

Были сформулированы требования, предъявляемые к разрабатываемой архитектуре, уточняющие формулировку соответствующей частной задачи исследования. Для разработки обобщенной архитектуры был выбран мультиагентный подход. Основными требованиями, предъявляемыми к разрабатываемой архитектуре, являются масштабируемость, открытость и прозрачность.

На основе сформулированных требований была разработана обобщенная архитектура программного комплекса контроля целостности мультимедиа-информации. Данная архитектура может быть реализована на различных архитектурах создания распределенных систем, таких как DCOM, CORBA, Java/RMI. За счет использования специальных агентов, данная архитектура позволяет осуществлять гибкое масштабирование комплекса и его настройку в соответствии с уже имеющейся распределенной информационно-вычислительной инфраструктурой.

Разработанная архитектура состоит из программных агентов шести типов: агент контроля и управления представляет собой единый центр управления всем комплексом защиты мультимедиа-информации, агент информационного хранилища предназначен для организации распределенного хранения служебной и мультимедийной информации, агент управления предназначен для управления всеми агентами комплекса, агент регистрации мультимедиа-информации предназначен для внедрения ЦВЗ с кодом контроля целостности, агент клонирования мультимедиа-информации предназначен для предоставления пользователям копий мультимедиа-информации из хранилища с уникальным ЦВЗ, агент анализа и контроля целостности предназначен для поиска мультимедиа-информации, извлечения ЦВЗ, проверки его корректности и контроля целостности.

Приведены результаты разработки архитектуры агентов всех типов. Все агенты, входящие в состав комплекса, построены с использованием модульной архитектуры. В архитектуру всех агентов входит главный программный модуль и функциональные

модули, которые могут быть либо обязательными (необходимыми для штатной эксплуатации комплекса), либо необязательными (наличие которых необязательно для работы комплекса). Подобная архитектура агентов позволяет быстро изменять и наращивать их функциональность за счет разработки и подключения новых модулей.

Описаны универсальные интерфейсы взаимодействия управляющих и функциональных компонент агентов комплекса. Данные интерфейсы используют технологию динамически подключаемых библиотек. Их универсальность заключается в том, что они позволяют снизить сложность разработки агентов за счет подключения одних и тех же модулей к различным агентам. Кроме того, использование данных интерфейсов позволяет производить изменение используемых в агентах функциональных модулей непосредственно во время работы комплекса.

Разработаны и описаны открытые интерфейсы для подключения программного обеспечения стороннего разработчика, а именно интерфейс для подключения модулей для работы с мультимедиа объектами IMediaStream, интерфейс для подключения модулей выделения пространства сокрытия IMediaContainer, интерфейс для подключения модулей работы с массивами мультимедиа потоков IArrayOfMediaStream. Данные интерфейсы позволяют подключать модули, отвечающие за выделение пространства сокрытия и работу с различными форматами хранения мультимедиа-информации. Они построены при помощи механизма СОМ интерфейсов, что позволяет упростить процесс разработки и отладки модулей сторонними разработчиками.

Рассмотрен вопрос практической реализации разработанной архитектуры программного комплекса контроля целостности мультимедиа-информации с учетом разработанной модели естественности и методов внедрения и извлечения ЦВЗ. На основе разработанной архитектуры программного обеспечения, математических моделей и методов для демонстрации возможности реализации предлагаемых принципов была разработана программная система контроля целостности мультимедиа-информации.

Разработанные программные средства предназначены для организации распределенного хранения мультимедиа-информации в информационно-телекоммуникационных системах и контроля ее целостности и построены с использованием функциональных модулей, использующих как универсальные интерфейсы взаимодействия управляющих и функциональных компонент, так и открытые интерфейсы для подключения программного обеспечения стороннего разработчика.

Проведена экспериментальная оценка аспектов функционирования комплекса, вносимых при сокрытии искажений и нарушению естественности мультимедиаконтейнеров, которая позволила определить максимально возможную плотность сокрытия, не приводящую к внесению статистических искажений. Анализ, проведенный при помощи разработанной модели оценки перцептивных искажений, показал что искажения вносимые при сокрытии с допустимой с точки зрения нарушения естественности плотностью сокрытия, не нарушают перцептивное качество контейнера.

ЛИТЕРАТУРА

1. Bender W., Gruhl D., Morimoto N.: "Method and apparatus for echo data hiding in audio signals", United States Patent № 5,893,067, April 06, 1999, US Patent & Trademark Office.
2. Bender W., Morimoto N., Gruhl D.: "Method and apparatus for data hiding in images", United States Patent № 5,870,499, February 09, 1999, US Patent & Trademark Office.
3. Bender W., Morimoto N., Gruhl D.: "Method and apparatus for logo hiding in images", United States Patent № 6,201,879, March 13, 2001, US Patent & Trademark Office.
4. Bolle R.M., Connell J.H., Ratha N.K.: "System and method for data hiding in compressed fingerprint images", United States Patent № 6,301,368, October 09, 2001, US Patent & Trademark Office.
5. Cachin C. An information-theoretic model for steganography // Proceedings of 2nd International Workshop on Information Hiding. 1998. P. 306-318.
6. Chang Shih-Fu; Meng Jianhao: "Watermarking of digital image data", WO9922480 (EP1034635), May 06, 1999, World Intellectual Property Organization.
7. Cho Jung Seok, Kim Jong Weon, Choi Jong Uk, Shin Seung Won (KR): "Digital watermarking method and apparatus using a color image as a watermark", WO0188883, November 22, 2001, World Intellectual Property Organization.
8. DeMont J.P.: "Method for hiding a binary encoded message in an electronic document by modulating the case of the characters in a case-insensitive markup language", United States Patent № 5,920,878, July 06, 1999, US Patent & Trademark Office.
9. Fridrich J., Simard R.J.: "Secure encryption and hiding of data and messages in images", United States Patent № 6,094,483, July 25, 2000, US Patent & Trademark Office.
10. Hady R.A. Provable security by one-way functions // РГУ, физфак, 2003, Федеральная целевая программа "Интеграция", 55-я студенческая научная конференция физического факультета, April 2003 стр. 42.
11. Ho Anthony Tung Shuen, Tam Siu Chung, Yap Lian Teck, Tan Siong Chai (SG): "Methods of digital steganography for multimedia data", WO0025203 (AU6494399), May 04, 2000, World Intellectual Property Organization.
12. International standart ISO/IEC 14882:1998. Programming language – C++.

13. Johnson Neil F., Jajodia Sushil: "Exploring steganography: seen the unseen", IEEE Computer – February 1998, 26-34.
14. Johnson Neil F., Jajodia Sushil: "Exploring steganography: seen the unseen", IEEE Computer – February 1998, 26-34.
15. Massey J.L. Contemporary cryptography: An introduction, Contemporary Cryptology // The Science of Information Integrity. IEEE Press, 1991. P. 1-39.
16. Moskowitz S.A., Cooperman M.: "Method and system for digital watermarking", United States Patent № 5,905,800, May 18, 1999, US Patent & Trademark Office.
17. Moskowitz S.A., Cooperman M.: "Method for stega-cipher protection of computer code", United States Patent № 5,745,569, April 28, 1998, US Patent & Trademark Office.
18. Moskowitz S.A., Cooperman M.: "Optimization methods for the insertion, protection, and detection of digital watermarks in digitized data", United States Patent № 5,889,868, March 30, 1999, US Patent & Trademark Office.
19. Moskowitz S.A., Cooperman M.: "Steganographic method and device", United States Patent № 5,687,236, November 11, 1997, US Patent & Trademark Office.
20. Moskowitz S.A.: "Multiple transform utilization and applications for secure digital watermarking", United States Patent № 6,205,249, March 20, 2001, US Patent & Trademark Office.
21. Moskowitz S.A.: "Z-transform implementation of digital watermarks", United States Patent № 6,078,664, June 20, 2000, US Patent & Trademark Office.
22. Petrovic R.: "Apparatus and method for embedding and extracting information in analog signals using replica modulation", WO0000969 (EP1095376), January 06, 2000, World Intellectual Property Organization.
23. Pfitzmann B. Information Hiding Terminology // Information hiding: first international workshop, Lecture Notes in Computer Science. Springer, 1996. Vol. 1174. P. 347-350.
24. Rhoads G.B.: "Audio or video steganography", United States Patent № 6,266,430, July 24, 2001, US Patent & Trademark Office.
25. Rhoads G.B.: "Graphics processing system employing embedded code signals", United States Patent № 5,768,426, June 16, 1998, US Patent & Trademark Office.

26. Rhoads G.B.: "Image steganography system featuring perceptually adaptive and globally scalable signal embedding", United States Patent № 5,748,763, May 05, 1998, US Patent & Trademark Office.
27. Rhoads G.B.: "Methods for detecting alteration of audio and images", United States Patent № 6,289,108, September 11, 2001, US Patent & Trademark Office.
28. Rhoads G.B.: "Methods for optimizing watermark detection", United States Patent № 6,307,949, October 23, 2001, US Patent & Trademark Office.
29. Rhoads G.B.: "Network linking method using steganographically embedded data objects", United States Patent № 5,841,978, November 24, 1998, US Patent & Trademark Office.
30. Rhoads G.B.: "Photographic products and methods employing embedded information", United States Patent № 5,822,436, October 13, 1998, US Patent & Trademark Office.
31. Rhoads G.B.: "Signal processing to hide plural-bit information in image, video, and audio data", United States Patent № 6,122,392, September 19, 2000, US Patent & Trademark Office.
32. Rhoads G.B.: "Steganographic methods and media for photography", United States Patent № 6,111,954, August 29, 2000, US Patent & Trademark Office.
33. Rhoads G.B.: "Steganographic system", United States Patent № 5,850,481, December 15, 1998, US Patent & Trademark Office.
34. Rhoads G.B.: "Steganography methods employing embedded calibration data", United States Patent № 5,636,292, June 03, 1997, US Patent & Trademark Office.
35. Rhoads G.B.: "Steganography systems", WO9636163 (AU6022396, CA2218957, JP2002504272T), November 14, 1996, World Intellectual Property Organization.
36. Rhoads G.B.: "Video steganography", United States Patent № 6,026,193, February 15, 2000, US Patent & Trademark Office.
37. Rhoads G.B.: "Wireless telephony with steganography", United States Patent № 6,278,781, August 21, 2001, US Patent & Trademark Office.
38. Sandford II, Maxwell T., Handel T.G., Bradley J.N.: "Compression embedding", United States Patent № 5,778,102, July 07, 1998, US Patent & Trademark Office.
39. Simmons G.J. An introduction to shared secret and/or shared control schemes and their application // Contemporary Cryptology: The Science of Information Integrity. IEEE Press, 1991. P. 441-497.

40. Simmons G.J. Authentication theory/coding theory // Advances in Cryptology: Proceedings of Crypto 84. Lecture Notes in Computer Science. Springer, 1985. Vol. 196.
41. Simmons G.J. The prisoners' problem and the subliminal channel // Advances in Cryptology: Proceedings of Crypto 83. Plenum Press, 1984. P. 51-67.
42. Van Wie D.M., Weber R.P.: "Steganographic techniques for securely delivering electronic digital rights management control information over insecure communication channels", United States Patent № 6,240,185, May 29, 2001, US Patent & Trademark Office.
43. Zollner J., Federrath H., Klimant H., Pfitzmann A., Piotraschke R., Westfeld A., Wicke G., Wolf G. Modeling the security of steganographic systems // Proc. 2nd Workshop on Information Hiding. Lecture Notes in Computer Science. Springer, 2000.
44. Аграновский А.В., Балакин А.В. Стеганография в тексте // Труды конференции Безопасность информационных технологий, том 2, Пенза, 2001, стр. 15-16.
45. Аграновский А.В., Балакин А.В., Хади Р.А. Интеллектуальные стеганографические системы // Искусственный интеллекст - 2002, материалы международной научно-технической конференции, п. Кацивели, Крым, Украина, 16-20 сентября, том 2, Таганрог: Изд-во ТРТУ, 2002, стр. 88-90.
46. Аграновский А.В., Балакин А.В. Мультиагентная архитектура систем стеганоанализа // ИМС-2003, Материалы Международной научной конференции. Т.2. Таганрог: Изд-во ТРТУ, 2003, стр. 33-35.
47. Аграновский А.В., Балакин А.В., Рутковский Н.В. Мультиагентный подход к решению задач стеганографического анализа // Научно-теоретический международный журнал "Искусственный интеллеккт", N3, Донецк: Дон ГИИИ, Украина, 2003, стр. 428-431.
48. Аграновский А.В., Балакин А.В., Рутковский Н.В., Хади Р.А. Контроль искажений растровых графических изображений, вносимых стеганографическими системами // Материалы Всероссийской научно-технической конференции с международным участием «Компьютерные технологии в инженерной и управлении деятельности», Таганрог, изд-во ТРТУ, 2003, Известия ТРТУ №3(32),стр.16-18.
49. Аграновский А.В., Балакин А.В., Рутковский Н.В., Хади Р.А. Методология противодействия современным системам стеганоанализа // V Межд. конференция "РусКрипто-2003" по современной криптологии, стеганографии, цифровой

подписи и системам защиты информации, электронный сборник докладов Ассоциации "РусКрипто".

50. Аграновский А.В., Балакин А.В., Рутковский Н.В., Хади Р.А. Теоретико-информационный подход к оценке стойкости стеганографических систем // Методы и алгоритмы прикладной математики в технике, медицине и экономике: Материалы III Международной научно-практической конференции, г.Новочеркасск: ЮРГТУ, 2003. – Ч. 2. – с.50-52.
51. Аграновский А.В., Балакин А.В., Хади Р.А, Котов И.Н. Анализ информационной безопасности с представлением на математических графах // Известия ТРТУ. Тематический выпуск: Материалы всероссийской научно-технической конференции с международным участием "Компьютерные технологии в инженерной и управлеченческой деятельности", Таганрог, ТРТУ, №2 (25), 2002, стр. 240-244.
52. Аграновский А.В., Балакин А.В., Хади Р.А. Современные запатентованные решения в области стеганографии // Телекоммуникации, № 1, 2003, с. 13-19.
53. Аграновский А.В., Балакин А.В., Хади Р.А. Визуальная криптография: технология и угрозы// Проблемы управления безопасностью сложных систем: Труды X международной конференции. Москва, декабрь 2002 г. // под ред. Н.И.Архиповой и В.В.Кульбы. Часть 1. – М.: РГГУ – Издательский дом МПА-Пресс, стр. 316-318.
54. Аграновский А.В., Балакин А.В., Хади Р.А. Запатентованные методы стеганографии в технологиях цифровых водяных знаков // Информационные технологии, №9, М.: Машиностроение, 2002, стр. 2-7.
55. Аграновский А.В., Балакин А.В., Хади Р.А. Защита информации в карманных персональных компьютерах // М:Солон-Пресс, Оргтехника и компьютеры, №2, стр. 39-46, 2003.
56. Аграновский А.В., Балакин А.В., Хади Р.А. Классические шифры и методы их криptoанализа // Информационные технологии, №10, М.: Машиностроение, 2001 стр. 40-45.
57. Аграновский А.В., Балакин А.В., Хади Р.А. Обучаемые системы стеганографии // Искусственный интеллект, №4, Донецк, Украина, 2002, стр. 132-135.
58. Аграновский А.В., Балакин А.В., Хади Р.А. Организация связи с помощью методов визуальной криптографии // Материалы региональной научно-практической

конференции "Проблемы информационной безопасности и защиты информации", ТГУ, Тула, 2002, стр. 34-36.

59. Аграновский А.В., Балакин А.В., Хади Р.А. Противодействие методам стеганографического сокрытия информации // Научный сервис в сети Интернет: Труды Всероссийской научной конференции (22-27 сентября 2003г., г.Новороссийск)-М.:Изд-во МГУ, 2003, стр.215-216.
60. Аграновский А.В., Балакин А.В., Хади Р.А. Современные подходы к уничтожению информации, сокрытой при помощи стеганографических систем // сборник тезисов конференции "Дагинформ-2003", с.67-68, 2003.
61. Аграновский А.В., Балакин А.В., Хади Р.А., Котов И.Н. Интеллектуальные системы обнаружения и защиты от вторжений // ИМС-2001, Тезисы докладов международной конференции, Дивноморское, Россия, 1-6 октября, Таганрог: Изд-во ТРТУ, 2001, стр. 307-309.
62. Аграновский А.В., Девягин П.Н., Черемушкин А.В., Хади Р.Л. Основы стеганографии. М: Радио и связь, 2003, 192 стр.
63. Аграновский А.В., Жижелев А.В., Хади Р.А., Балакин А.В. Оценка уровня скрытности встраивания данных в стеганографических системах первого поколения // Шестая Международная Конференция "Комплексная защита информации", ВНИИПВТИ, Москва, 2002.
64. Аграновский А.В., Репалов С.А., Рутковский Н.В., Хади Р.А. Стеганографические методы внедрения информации в аудио-сообщения // Научная сессия МИФИ-2003. X Всероссийская научная конференция «Проблемы информационной безопасности в системе высшей школы». Сборник научных трудов. М.: МИФИ, 2003. стр. 15-16.
65. Аграновский А.В., Хади Р.А., "Практическая криптография: алгоритмы и их программирование", М: Солон-Пресс, 2002 – 256 стр.
66. Аграновский А.В., Хади Р.А., Балакин А.В. Защита информации в сетях Bluetooth // Журнал информационных технологий – Чип (Chip), № 12, 2003, стр.112-117.
67. Аграновский А.В., Хади Р.А., Балакин А.В. Обучаемые системы обнаружения и защиты от вторжений // Искусственный интеллект, N3, Донецк, Украина, 2001, стр. 440-444.

68. Аграновский А.В., Хади Р.А., Балакин А.В. Эвристическое доказательство стойкости крипtosистем // Труды конференции Безопасность информационных технологий, том 2, Пенза, 2001, стр. 17-18.
69. Аграновский А.В., Хади Р.А., Балакин А.В., Котов И.Н. Информационная безопасность удаленного доступа в сети Интернет // Труды международной научно-методической конференции Телематика, русский том, СПб, июнь 2001, стр. 40.
70. Аграновский А.В., Хади Р.А., Балакин А.В., Мартынов А.П., Николаев Д.Б., Фомченко В.Н. Разработка архитектуры программного комплекса информационной защиты ведомственной локальной сети и рабочих станций, Учебно-методической пособие. – Саров: "ИНФО", 51с.:ил.
71. Аграновский А.В., Хади Р.А., Балакин А.В., Репалов С.А. Scripher: криптографическая защита исполнимых исходных текстов на языках Perl и C++/PerlXS // Свидетельство об официальной регистрации программы для ЭВМ №2003611307/РОСПАТЕНТ. - М., 29.05.2003.
72. Аграновский А.В., Хади Р.А., Балакин А.В., Репалов С.А. Программный комплекс, предназначенный для стеганографического сокрытия информации в текстовых сообщениях // Свидетельство об официальной регистрации программы для ЭВМ №2003611313/РОСПАТЕНТ. - М., 29.05.2003.
73. Аграновский А.В., Хади Р.А., Ерусалимский Я.М., "Криптография и открытые системы", М: Машиностроение, Телекоммуникации, N2, 2001 г., стр. 13 – 23.
74. Алексеев В.М., Андрианов В.В., Зефиров С.Л., Лупанов И.Ю. Комплекс защиты информации для автоматизированных информационных систем с использованием баз данных. // Безопасность информационных технологий. -1994, №3-4, с. 128-130.
75. Алиев А.Т., Балакин А.В. К вопросу оценки стойкости стегосистем // Методы и алгоритмы прикладной математики в технике, медицине и экономике: Материалы V Междунар. науч.-практ. конф., г. Новочеркасск, 21 янв. 2005 г.: В 3 ч./Южно-Рос. гос. техн. ун-т (НПИ). - Новочеркасск: ЮРГТУ, 2004. - Ч. 1. с. 51-53.
76. Алиев А.Т., Балакин А.В. Оценка стойкости систем скрытой передачи информации // Известия ТРТУ. Тематический выпуск., Изд-во ТРТУ, 2005. №4 (48), с. 199-204.
77. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии. М.: Гелиос АРВ, 2001.

78. Артехин Б.В. Стеганография // Защита информации. Конфидент, №4, 1996, с. 47-50.
79. Балакин А.В. Защита информации методами стеганографии в образовательных системах // тезисы докладов, научно-методическая конференция "Современные информационные технологии в образовании: Южный Федеральный округ", г.Ростов-на-Дону,2003г., стр.31-34.
80. Балакин А.В. Использование методов уничтожения сокрытой информации в задачах стеганоанализа, Методы и алгоритмы прикладной математики в технике, медицине и экономике: Материалы IV Междунар. науч.-практ. конф., г.Новочеркасск, 2004 г.: В 4 ч. / Юж.-Рос. гос. техн. ун-т (НПИ). – Новочеркасск: ЮРГТУ, 2004. – Ч.1, с.43-44.
81. Балакин А.В. Модель контроля тональности в задачах стеганографии // Моделирование. Теория, методы и средства: Материалы III Международной научно-практической конференции, г.Новочеркасск, 11 апреля 2003г.: В 5 ч. / ЮРГТУ. – Новочеркасск: ЮРГТУ, 2003. – Ч. 3. – стр. 48-50.
82. Балакин А.В. О границах применимости моделей естественности для анализа мультимедиа-информации // Материалы шестого Всероссийского симпозиума по прикладной и промышленной математике, т.12, выпуск 2, с. 297-298, 2005.
83. Балакин А.В., Полупкин Н.Ю. Особенности построения программно-аппаратной подсистемы сбора данных об окружающей физической среде в интеллектуальной системе обнаружения и защиты от вторжений // Искусственный интеллект. Интеллектуальные и многопроцессорные системы-2004 // Материалы Международной научной конференции. Т.1. Таганрог: Изд-во ТРТУ, 2004, стр. 340-342.
84. Балакин А.В., Репалов С.А, Шагов Г.Н. Современная стеганография: модели и методы преобразования информации. – Ростов-на-Дону:Изд-во СКНЦ ВШ, 2004. 240 с.:ил.
85. Балакин А.В., Рутковский Н.В. Игровые модели в задачах стеганоанализа // Моделирование. Теория, методы и средства: Материалы III Международной научно-практической конференции, г.Новочеркасск, 11 апреля 2003г.: В 5 ч. / ЮРГТУ. – Новочеркасск: ЮРГТУ, 2003. – Ч. 3. – стр. 50-52.

86. Барсуков В.С. Безопасность: технологии, средства, услуги. - М.: Кудиц - Образ, 2001. - 496 с.
87. Барсуков В.С., Водолазкий В.В Современные технологии безопасности. - М.: "Нолидж", 2000, 496 с.
88. Барсуков В.С., Романцов А.П. Компьютерная стеганография, вчера, сегодня, завтра // "Специальная техника", № 4-5, 1998 г.
89. Барсуков В.С., Романцов А.П. Оценка уровня скрытности мультимедийных стеганографических каналов хранения и передачи информации // Специальная техника, № 6, 1999, с. 52-59.
90. Борисенко Б.Б. Внедрение водяных знаков, основанных на m-последовательностях, в графические изображения // Обозрение прикладной и промышленной математики, том 9, выпуск 1, М.:редакция журнала "ОПиПМ", 2002. - стр. 77-89.
91. Борин Д. Ю., Грибунин В. Г. Аутентификация сигналов методами стеганографии // II сессия молодежной школы-семинара "Экологическая и промышленная безопасность", Тезисы докладов, РФЯЦ-ВНИИЭФ, Саров, 2002.- стр. 96- 102.
92. Борин Д.Ю., Гончаров С.Н. Математические вычисления в стеганографии // Седьмая нижегородская сессия молодых ученых . (математические науки): тезисы докладов. – Н. Новгород: Изд. Гладкова О.В., 2002. – стр 5.
93. Быков С. Ф. Алгоритм сжатия JPEG с позиции компьютерной стеганографии // Защита информации. Конфидент, №3, 2000, стр. 26.
94. Быков С. Ф. Дискретное изображение: избыточность и внутренние детерминированные связи // Сборник тезисов докладов Российской научно-технической конференции "Методы и технические средства обеспечения безопасности информации". - СПб.: 2000, с. 93-95.
95. Быков С. Ф. Особенности защиты цифровой интеллектуальной собственности // Защита информации. Конфидент, №3, 2001, стр. 50-55.
96. Быков С.Ф., Мотуз О.В. Основы стегоанализа // Защита информации. Конфидснт, №3, 2000, стр. 38.
97. Гайкович В., Першин А. Безопасность электронных банковских систем. – М.: Изд-во компания "Единая Европа", 1993 г., 363 с.

98. Гение О.В. Основные положения стеганографии // Защита информации. Конфидент, №3, 2000, стр.20.
99. Герасименко В.А. Проблемы защиты данных в системах их обработки. // Зарубежная радиоэлектроника, 1989 г., №12, с. 5-21.
100. Гольденберг Л.М., Матюшкин Б.Д.: "Цифровая обработка сигналов", М.: Радио и связь, 1990.
101. Гончаров С. Н., Грибунин В. Г. Выбор преобразования сигналов в стеганографии // II сессия молодежной школы-семинара "Экологическая и промышленная безопасность", Тезисы докладов, РФЯЦ-ВНИИЭФ, Саров, 2002.– стр. 102-107.
102. Грибунин В.Г., Оков И.Н., Туринцев И.В., Головачев В.Ю., Коняев А.В. Компьютерная стеганография – под научной редакцией Хади Р.А., М:Солон-Пресс, 2003 – 240 стр.
103. Дворянкин С.В. Скрытная передача конфиденциальной информации в аудио сигналах и речи // "БДИ", №2 (30), 2000, с. 12-16.
104. Дворянкин С.В., Романцов А.П. Статистический метод стеганофонического анализа аудиофайлов // Тезисы докладов IV межрегионального научно-технического семинара "Применение пластиковых карт и защиты информации". - М.: МНТОРЭС им. А.С. Попова, 1999, с. 51-54.
105. Девягин П.Н., Михальский О.О., Правиков Д.И. и др. Теоретические основы компьютерной безопасности: Учеб. пособие для вузов. – М.: Радио и связь, 2000, 192 с.
106. Десятерик М.Н. Система создания ключевых дискет // Сборник трудов первого регионального научно-практического семинара "Информационная безопасность – Юг России". - Таганрог-Кисловодск, 1999г, с. 103-104.
107. Задираха В.К., Мельникова С.С., Борадавка Н.В. Спектральный алгоритм компьютерной стеганографии // Искусственный интеллект, № 3, Донецк, Украина, 2002.
108. Задираха В.К., Олесюк О.С. Компьютерная криптология. - Киев: 2002 - 504 с.
109. Клевцов В.В., Шумовский Е.Ю., Балакин А.В. Инструментальные программные средства SHid-Tools для скрытного информационного взаимодействия с информационными системами в интерактивном режиме // Свидетельство об

официальной регистрации программы для ЭВМ №2005610537/РОСПАТЕНТ. - М., 28.02.2005.

110. Клопов В.А., Мотуз О.В. Основы компьютерной стеганографии // Защита информации. Конфидент, №4, 1997, с. 43-48.
111. Козина Г.Л., Михайленко А.Г. Система "CRYPT&HIDE" шифрования и встраивания информации для передачи по открытому каналу связи // Методы и технические средства обеспечения безопасности информации: Тезисы докладов. - СПб.: Издательство СПбГПУ, 2002, с. 82-84.
112. Копытков В.В. Маскировка побочных каналов излучений методом статистически необратимых преобразований сигнала // Сборник трудов научно-практической конференции "Информационная безопасность". – Таганрог: Издательство ТРТУ, 2001, с. 120-121.
113. Курочкин А. А., Грибунин В. Г. Теоретико-сложностный подход к оценке стойкости Стегосистем // II сессия молодежной школы-семинара "Экологическая и промышленная безопасность", Тезисы докладов, РФЯЦ-ВНИИЭФ, Саров, 2002.- стр. 107-112.
114. Кустов А. Н., Федчук А. А. Методы встраивания скрытых сообщений // Защита информации. Конфидент, №3, 2000, стр. 34.
115. Липовецкий В.Ю. Исследование шумов видео применительно к задачам стеганографии // Научная сессия МИФИ-2003. X Всероссийская научная конференция "Проблемы информационной безопасности в системе высшей школы". Сборник научных трудов. – М.: МИФИ, 2003, стр. 123-125.
116. Мартынов А.А., Николаев Д.Б., Хади Р.А. Примеры простейших криптографических систем // Седьмая нижегородская сессия молодых ученых (математические науки): тезисы докладов. – Н. Новгород: Изд. Гладкова О.В., 2002. - стр. 15.
117. Оков И.Н., Ковалев Р.М. Электронные водяные знаки как средство аутентификации передаваемых сообщений // Защита информации. Конфидент, №3, 2001, стр. 50-55.
118. Патент РФ. Способ криптографического преобразования двоичных данных, № 2226041 HL/09. Аграновский А.В., Хади Р.А., Балакин А.В., Фомченко В.Ф.,

Мартынов С.П. Заявлено 02.06.2003; Положительное решение о выдаче патента по заявке № 2001129345/09 (031482) от 06.11.2003.

119. Патент РФ. Способ стеганографического преобразования блоков двоичных данных, Аграновский А.В., Балакин А.В., Репалов С.А., Хади Р.А., №2257010 ФИПС, 20 июля 2005. Заявлено 27.03.2002; Приоритет № 2002107479/09 от 27.03.2002.
120. Патент РФ. Способ шифрования блоков данных, Аграновский А.В., Хади Р.А., Балакин А.В., №2207736, РОСПАТЕНТ, по заявке №2001110662, дата поступления 20.04.2001, приоритет от 20.04.2001.
121. Растиоргуев С.П. Исследование систем защиты информации. // НТИ. Сер. 2. Информационные процессы и системы., 1993, №12, с. 10-15.
122. Роджерсон Д. Основы СОМ. Пер. с англ. -М.: Издательский отдел "Русская редакция", ТОО "Channel Trading Ltd.", 1997.
123. Скаев Ю.И. Стеганографические методы защиты данных // Сборник трудов научно-практической конференции "Информационная безопасность". – Таганрог: Издательство ТРТУ, 2000, с. 52-53.
124. Спесивцев А.В., Вегнер В.А., Крутяков А.Ю., Серегин В.В., Сидоров В.А. Защита информации в персональных ЭВМ - М.: Радио и связь, "Веста", 1992, 191 с.
125. Стельмах Э.П. Стеганографический анализ // Сборник трудов X международной научной конференции "Информатизация правоохранительных систем". - Москва, 2001, с. 371-372.
126. Страуструп Б. Язык программирования C++. Специальное издание. - М.: БИНОМ, 2004
127. Харрис Л. Программирование OLE. Освой самостоятельно за 21 день. Пер. с англ. – М.: БИНОМ, 1995.
128. Хомяков Е.И., Федоров В.М., Макаревич О.Б. Стеганография: применение и обнаружение // Сборник трудов научно-практической конференции "Информационная безопасность"/ - Таганрог: Издательство ТРТУ, 2000, с. 50-52.
129. Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес Приемы объектно-ориентированного программирования. Паттерны проектирования. – СПб: Питер, 2001.

ПРИЛОЖЕНИЯ

Приложение А. Обзор запатентованных решений в области внедрения цифровых водяных знаков

Общий метод сокрытия данных в оцифрованных сигналах, разработанный Джейфри Роадсом, сотрудником корпорации Digimarc Corporation (специализирующейся на разработке методов внедрения цифровых водяных знаков в оцифрованные данные и практическом применении данных технологий), описан в патенте США “Стеганографическая система” [33] и в международном патенте “Стеганографические системы” [35].

Патентуемые решения базируются на том факте, что оцифрованные данные содержат так называемые шумы квантования. Данные шумы возникают в процессе оцифровки за счет того, что непрерывное значение амплитуды аналогового сигнала аппроксимируется дискретными значениями квантования. Замена имеющихся в контейнере шумов на сигнал, уровень которого не превышает уровень шумов, позволяет скрыть сообщение так, что при визуальном контроле модифицированный контейнер будет неотличим от исходного контейнера.

Заявленное в патенте Роадса стеганографическое преобразование состоит в том, что на основе сообщения и секретного ключа генерируется псевдослучайный сигнал, уровень которого не превышает уровень шумов пустого контейнера. С помощью фильтрации (применения фильтров) шум пустого контейнера удаляется, и результат суммируется с псевдослучайным сигналом. Применение данного метода для сокрытия двоичных сообщений в растровых графических изображениях, цифровом звуке и цифровом видео описывается в патентах: “Аудио- и видеостеганография” [24], “Обработка сигналов для сокрытия информации в графических, видео- и аудиоданных” [31], “Видеостеганография” [36], “Методы для определения изменений в аудио сигналах и графических изображениях” [27].

Скотт Москвиц и Марк Куперман разработали оригинальный метод, описанный в патенте “Стеганографический метод и устройство” [19]. В нем, так же как и в разработке предыдущего автора, в качестве контейнеров выбраны оцифрованные данные произвольной природы. Но стеганографическое преобразование заключается в том, что исходное сообщение предварительно шифруется с использованием секретного ключа. Контейнер разбивается на блоки с длиной равной 2^n , к которым применяется дискретное преобразование Фурье. На основе секретного стеганографического ключа с помощью

генератора псевдослучайных чисел вычисляется ключевая последовательность заданной длины. В данном контексте ключевая последовательность носит несколько иной характер, нежели используемая в криптографии. После ее выработки, младшие биты тех отсчетов очередного блока спектра, которым соответствуют единичные биты стеганографической ключевой последовательности, заменяются на очередные биты зашифрованного сообщения. Таким образом, сокрытие данных происходит не в отсчетах сигнала-контейнера, а в амплитудах гармоник его спектра.

В международном патенте Рэйда Петровика “Метод и аппарат для внедрения и извлечения информации из аналоговых сигналов с использованием повторной модуляции” [22] в качестве сообщений могут быть выбраны как цифровые, так и аналоговые данные, а в качестве контейнеров используются аналоговые сигналы (например, аудио- и видеозаписи). Сокрытие состоит в том, что на основе немодифицированного сигнала-контейнера и исходного сообщения генерируются повторные сигналы, которые суммируются с контейнером. Прямое стеганографическое преобразование состоит из трех последовательных преобразований:

- генератора (генерирующего повторные сигналы на основе немодифицированного контейнера и секретного ключа);
- модулятора (изменяющего повторные сигналы в зависимости от исходного сообщения);
- сумматора (добавляющего модифицированные повторные сигналы в немодифицированный контейнер).

Извлечение сокрытого сообщения из модифицированного контейнера состоит в генерации повторных модифицированных сигналов и их корреляции с контейнером. Обратное преобразование состоит так же из трех составляющих:

- генератора (генерирующего повторные сигналы на основе модифицированного контейнера и секретного ключа);
- модулятора (изменяющего модифицированный контейнер в зависимости от повторных сигналов);
- экстрактора (извлекающего исходное сообщение из измененного модифицированного контейнера путем применения специальных фильтров).

Стеганографическая система, разработанная сингапурскими учеными, описана в международном патенте “Методы цифровой стеганографии в мультимедийных данных”

[11]. Она предназначена для организации скрытых каналов передачи секретных данных и использует в качестве контейнеров цифровую мультимедийную информацию. Работа системы базируется на разработанном авторами генераторе псевдослучайных чисел, сокрытие происходит с использованием примитивных криптографических преобразований (сдвигов, перестановок и т.п.). Однако, поскольку авторы не привели формального доказательства стойкости предложенных преобразований, уровень скрытности предложенной системы пока остается под вопросом.

В патенте “Шифрование и сокрытие данных и сообщений в графических изображениях” [9] сообщениями являются растровые графические изображения с размером m на n точек. В качестве контейнера используется изображение с размером $2m$ на $2n$. Стеганографическое преобразование состоит в том, что исходное сообщение шифруется с использованием секретного ключа. Контейнер разбивается на блоки с размером 2 на 2 точки. Между полученными блоками и точками сообщения устанавливается однозначное соответствие согласно их координатам. Сокрытие происходит одинаково для всех точек и цветовых составляющих (RGB) и состоит в том, что 8 бит, характеризующие цветовую интенсивность, разбиваются на 4 пары по 2 бита. Первая пара согласно таблице:

2 бита	Приращение
00	+1
01	0
10	-1
11	-2

изменяет значение соответствующей цветовой характеристики верхней левой точки соответствующего блока контейнера, вторая – правой верхней и т.д. Таким образом, сокрытие происходит в младших битах. Для извлечения встроенного сообщения необходимо иметь секретный ключ и пустой контейнер, что затрудняет использование на практике данного метода. Похожий по идеи использовать для сокрытия младшие биты метод описан в патенте “Метод и аппарат для сокрытия данных в графических изображениях” [2]. Он состоит в том, что на основе секретного ключа формируется цепочка неповторяющихся координат точек графического изображения, выбранного в качестве контейнера. Исходное сообщение скрывается в младших битах интенсивностей цветовых составляющих, отвечающих точкам полученной цепочки. Однако, после применения специального фильтра, факт использования младших бит для сокрытия данных возможно обнаружить даже с помощью визуального контроля (см. рис. 3 и 4).

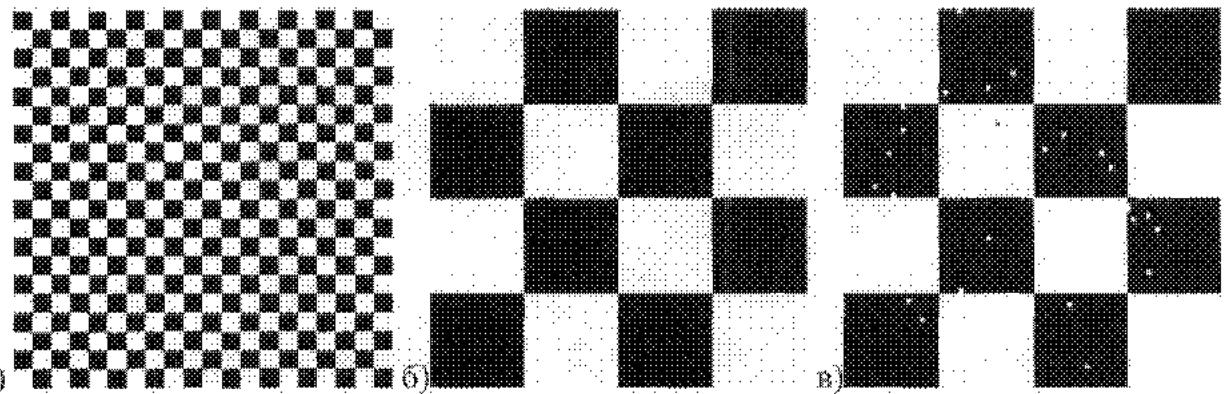


Рисунок 54 – а) не модифицированное изображение, б) фрагмент не модифицированного изображения после применения фильтра, в) фрагмент модифицированного изображения после применения фильтра

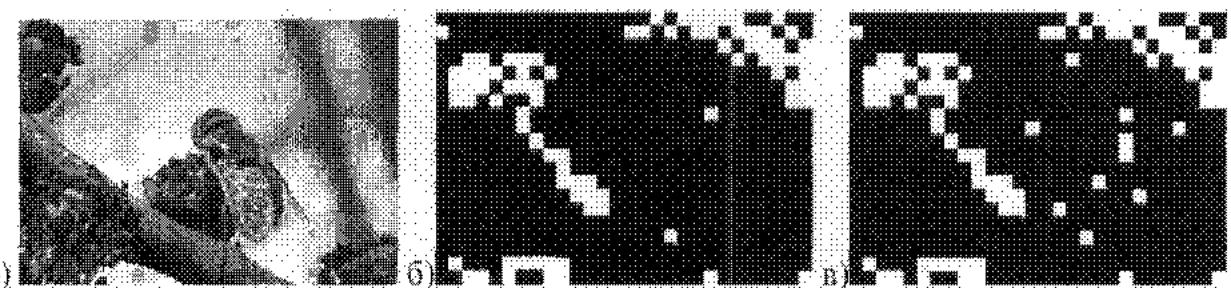


Рисунок 55 – а) не модифицированное фотографическое изображение, б) фрагмент не модифицированного изображения после применения фильтра, в) фрагмент модифицированного изображения после применения фильтра

Немалый практический интерес представляют методы, использующие в качестве контейнеров графические изображения, сжатые с потерей информации. В патенте “Система и метод для сокрытия данных в сжатых изображениях отпечатков пальцев” [4] в качестве контейнеров используются растровые графические изображения, сжимаемые с помощью модифицированного алгоритма JPEG. Согласно патенту, сокрытие состоит в том, что в процессе сжатия, после квантования, младшие биты ненулевых значений элементов блоков изображения заменяются на биты сообщения согласно псевдослучайной ключевой последовательности. Оригинальные биты сохраняются в полях для комментариев, что позволяет сохранить исходное изображение, но сильно ухудшает уровень скрытности данного метода.

Еще один метод сокрытия в сжатых графических изображениях был разработан сотрудниками Калифорнийского университета и описан в патенте “Сжимающее

внедрение” [38]. Прямое стеганографическое преобразование, как и в предыдущем патенте, представляет собой модифицированный алгоритм сжатия JPEG, в котором после квантования с помощью модификации значений элементов блоков изображения, определяемой секретным ключом, исходное сообщение встраивается в сжатое графическое изображение. В отличие от предыдущего метода, элементы для модификации выбираются с учетом их значения и вероятности появления этого значения. Что позволяет минимизировать изменение исходной энтропии контейнера, а значит снизить влияние сокрытия на эффективность использования алгоритма Хаффмана. В патенте приведен пример реализации описанного метода в виде компьютерной программы.

Стеганографическая система, в которой в качестве контейнеров используется цифровой звук, была разработана в Массачусетском Институте Технологии и заявлена в патенте “Метод и аппарат для сокрытия данных в эхо аудиосигналов” [1]. Сообщение в данной системе скрывается в параметрах добавляемого к звуку эхо-сигнала (амплитуде, задержках и количестве повторений). Данный метод основан на том, что ухо человека не может различить незначительные изменения параметров добавляемого эхо-сигнала, используемые для сокрытия сообщения. Эхо воспринимается человеком, как привычный и присущий большинству звуков резонанс. Более того, отсутствие эхо-сигнала придает звуку некоторую нереальность (так называемый эффект бесконечного помещения). Сокрытие заключается в том, что к исходным аудио-данным добавляется эхо-сигнал, параметры которого зависят от внедряемого сообщения. Сообщение скрывается в двух параметрах эхо-сигнала: величине и задержке. Положительными сторонами методов сокрытия в младших битах и спектре, описанных в патенте The Dice Company “Оптимизированные методы для внедрения, защиты и детектирования цифровых водяных знаков в оцифрованных данных” [18], являются возможность применения кода коррекции ошибок и зависимость псевдослучайного сигнала ЦВЗ не только от ключа, но и от контейнера с целью увеличения надежности и защищенности встраиваемых ЦВЗ.

В другом патенте The Dice Company “Метод и система для внедрения цифровых водяных знаков” [16] описан иной способ увеличения надежности и защищенности встраиваемых ЦВЗ, который состоит в том, что параметры внедрения ЦВЗ регулируются пользователем, который оценивает искажения, возникающие за счет внедрения ЦВЗ. Таким образом, с учетом достаточной квалификации пользователя возможно внедрение ЦВЗ с максимальной эффективностью без внесения искажений заметных для человека.

В патенте “Стеганографическая система с глобальным распределением сигнала и адаптацией к восприятию” [26] рассматривается применение кодов исправления ошибок для увеличения устойчивости встроенных ЦВЗ, и приводятся реализации системы встраивания ЦВЗ для случаев фиксированного и иоточного типа контейнеров. Пример программного кода, реализующего встраивание и детектирование ЦВЗ, приведен в патенте “Стеганографический метод, с внедрением маркировочных данных” [34]. В патентах “Фотографические продукты и методы встраивания информации” [30] и “Стеганографические методы для фотографии” [32] рассматриваются методы применения разработанных технологий внедрения ЦВЗ к технологиям фото-производства. Рассмотренные методы позволяют внедрять ЦВЗ в негативные и позитивные фотографические слайды, а также в фотографии на фотобумаге, и применять детектирование в фотолабораториях с целью обеспечения авторских и имущественных прав на фотографии (защита от несанкционированного копирования и тиражирования фотоизображений).

В патенте “Использование и применение многократных преобразований для защиты цифровых водяных знаков” [20] описан метод встраивания двоичного сообщения в оцифрованные сигналы. Согласно описанному методу к блокам сигнала применяется преобразование Фурье, после чего на основе битов ключа для текущего блока сигнала определяются амплитуды гармоник, изменением которых кодируется исходное двоичное сообщение: нулевые биты сообщения не изменяют значение амплитуды гармоники, а единичные уменьшают ее на определенное пользователем значение. Исходные амплитуды, формируют ключ, предназначенный для извлечения внедренных данных и восстановления исходного сигнала.

Применение Z-преобразования для встраивания и извлечения двоичных водяных знаков в цифровой поток описано в патенте “Применение Z-преобразования для цифровых водяных знаков” [21]. Существуют различные преобразования оцифрованных сигналов (преобразование Фурье, различного рода свертки и т.д.), одним из них является Z-преобразование [100]:

$$X(z) = \sum_{n=-\infty}^{+\infty} x(n)z^{-n}, \text{ где}$$

$x(n)$ – дискретный амплитудно-временной сигнал,

$X(z)$ – плоское комплексное представление,

z – комплексная переменная,

В патенте предлагается использовать Z-преобразование для встраивания водяных знаков, следующим образом. По дискретному амплитудно-временному сигналу $x(n)$ вычисляется его Z-преобразование $X(z)$. Затем с помощью привычных методов в контейнер $X(z)$ добавляются цифровые водяные знаки. После чего, применением обратного Z-преобразования, восстанавливается дискретный амплитудно-временной сигнал, содержащий цифровые водяные знаки.

В патенте “Методы для оптимизации детектирования цифровых водяных знаков” [28] приводится метод и схема алгоритма, позволяющего улучшить качество встраивания водяных знаков. А именно, позволяющего сделать их более устойчивыми к таким преобразованиям, как сжатие графических изображений, изменение разрешения, поворот на заданный угол. Примерная схема оптимизации состоит в том, что перед окончательным встраиванием цифровых водяных знаков производится анализ изменений происходящих после применения тех или иных преобразований к графическому изображению. На основе результатов данного анализа выбирается оптимальный способ встраивания, т.е. наиболее устойчивый ко всем рассмотренным преобразованием и производится окончательное встраивание водяных знаков.

Международный патент “Метод и аппарат внедрения цифровых водяных знаков использующий цветное изображение в качестве водяного знака” [7] разработан корейскими специалистами. Описанный в нем метод позволяет внедрять цифровые водяные знаки (цветное графическое изображение небольшого размера) в цифровой звук не внося искажения, заметные для человека при его прослушивании. Сокрытие состоит в том, что между отсчетами сообщения и контейнера устанавливается взаимнооднозначное соответствие. К контейнеру применяется wavelet-преобразование, а к сообщению дискретное косинусное преобразование. После этого полученные коэффициенты контейнера и сообщения комбинируются согласно установленному соответствию: коэффициенты сообщения умножаются на заданный коэффициент масштабирования (чем ближе его значение к нулю, тем меньше искажения, вносимые в контейнер, и ниже устойчивость внедренных ЦВЗ) и затем суммируются с коэффициентами контейнера. Применение обратного wavelet-преобразования к полученным после суммирования данным формирует модифицированный контейнер.

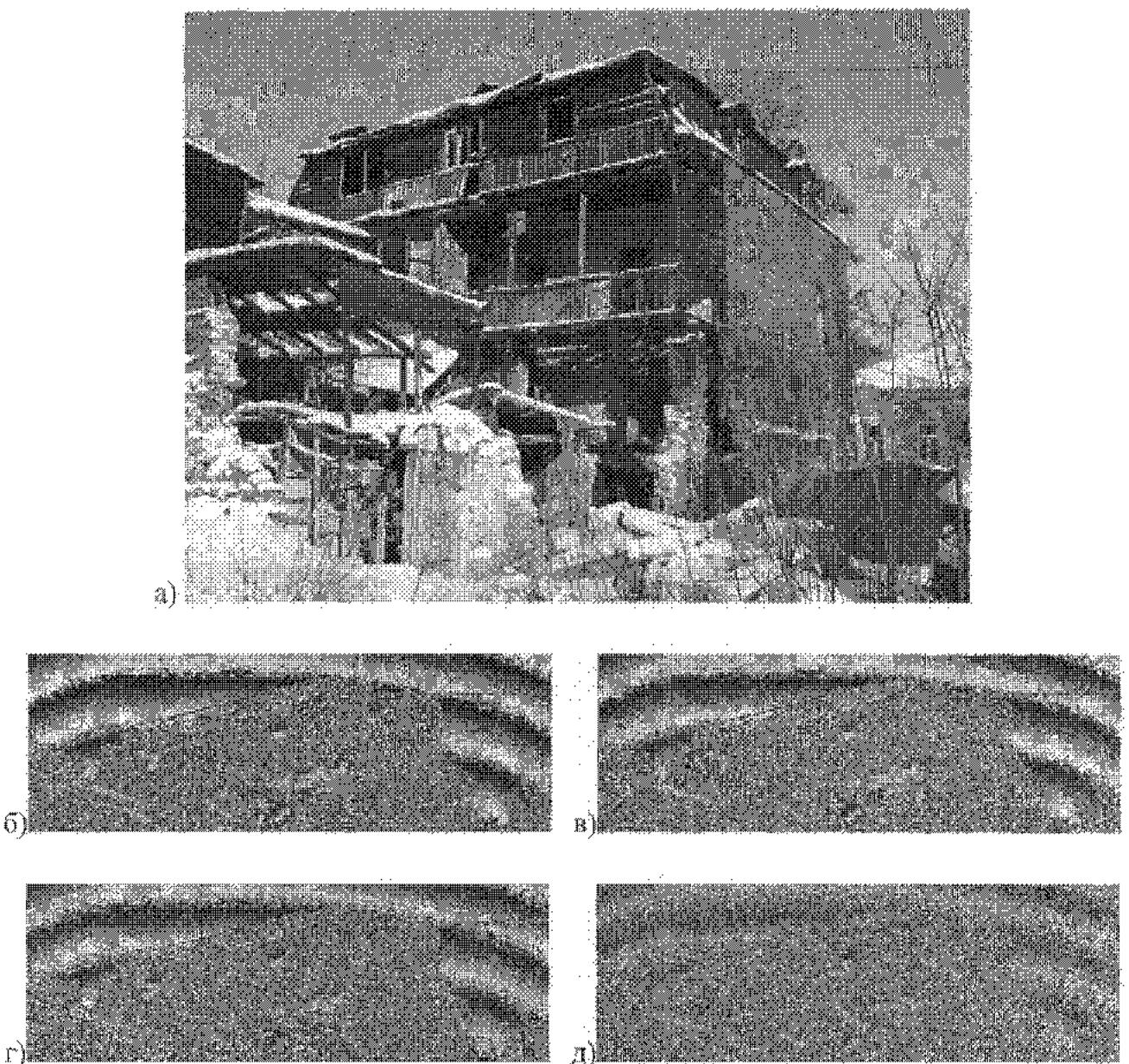


Рисунок 56 – Влияние уровня стойкости на искажения, происходящие за счёт внедрения ЦВЗ (а) немодифицированное изображение, б) фрагмент модифицированного изображения после применения фильтра (уровень стойкости 1), в) фрагмент модифицированного изображения после применения фильтра (уровень стойкости 2), г) фрагмент модифицированного изображения после применения фильтра (уровень стойкости 3), д) фрагмент модифицированного изображения после применения фильтра (уровень стойкости 4))

Метод, приведенный в патенте “Метод и аппарат для скрытия логотипов в графических изображениях” [3], позволяет скрывать простые графические изображения (логотипы, маски) в растровых графических файлах. Для скрытия логотипов (в данном контексте – произвольных простых геометрических форм одного цвета) в данном патенте

используется то, что зрение человека плохо различает высокочастотные изменения в графических изображениях. Скрытие состоит в том, что из одного высокочастотного участка изображения вырезается фрагмент с фигурой логотипа и копируется в другой не пресекающийся с исходным высокочастотный участок. Для извлечения скрытого логотипа необходимо обнаружить в изображении идентичные непересекающиеся подмножества. Недостатком предлагаемого метода является то, что в случае наличия пустого и полного контейнеров легко обнаружить скрытый логотип.

В международном патенте “Внедрение цифровых водяных знаков в цифровые графические изображения” [6] описан метод, позволяющий производить внедрение цифровых водяных знаков в цифровое видео изображение, сжатое с потерями информации с помощью алгоритма MPEG-2. В качестве ЦВЗ используются растровые графические изображения. Видеоизображение в формате MPEG-2 состоит из групп кадров (GOP), при этом в группу могут входить кадры трех различных типов: исходные, предсказуемые и двунаправленной интерполяции. Внедрение ЦВЗ состоит в том, что для каждой группы кадров с помощью специального преобразования создаются маски: изображение ЦВЗ преобразуется в черно-белое (отбрасываются цветоразностные составляющие), к нему применяется побочное дискретное косинусное преобразование, после чего производится масштабирование коэффициентов, зависящее от значений элементов блоков кадров текущей группы. После этого полученная маска суммируется с видеосигналом. Способ суммирования различен для исходных кадров, предсказуемых кадров и кадров двунаправленной интерполяции.

При решении ряда практических задач, возникает необходимость применения качественных алгоритмов позволяющих обеспечить внедрение цифровых водяных знаков с необходимым уровнем скрытности и защищенности. В частности, описанный в патенте “Метод для стего-крипто защиты программного кода” [17] метод служит для защиты программных продуктов от несанкционированного использования и копирования. Данный метод использует стеганографические системы для внедрения цифровых водяных знаков в звуковые и графические данные, находящиеся в программном коде, с целью их защиты от подмены и модификации.

В патенте “Стеганографические технологии для обеспечения защиты прав на электронную цифровую информацию, передаваемую в незащищенных каналах связи” [42] описываются методы и устройства, использующие технологии стеганографии (цифровые

водяные знаки) для управления доступом к информации, распространяемой по незащищенным каналам связи (к примеру, для организации телевизионного вещания).

В следующих патентах, разработанных сотрудниками Digimarc Corporation, описаны способы применения методов стеганографии (методы рассмотрены ранее) для решения практических задач. В патенте “Метод организации сетевого взаимодействия, основанный на применении стеганографически встроенных данных” [29] описывается применение сокрытия управляющей информации в распространяемых по сети графических, аудио и видео файлах с целью обеспечения сетевого взаимодействия, защиты авторских и имущественных прав, а также разграничения доступа. Для этого, перед размещением в сети, в графические, аудио и видео файлы встраивается идентификационная информация и цифровые водяные знаки. Идентификационная информация включает в себя информацию о передаваемых данных, об отправителе и ссылку на другой сетевой объект (аналог ссылки HTML). Авторами предлагается разработать протокол подобный протоколу HTTP, использующий информацию, внедренную подобным образом.

Представляют интерес методы управления различными системами и устройствами с помощью сокрытой информации. В патенте “Система обработки графических изображений с применением сокрытия кодовых сигналов” [25] описывается применение сокрытой информации в системах обработки изображений для обеспечения авторских и имущественных прав. Сокрытая информация определяет манипуляции, которые система обработки позволяет производить с тем или иным графическим изображением. К примеру, в изображение, готовое для печати, внедряется информация разрешающая только его печать (и запрещающая изменение). После чего изображение может быть передано для печати, без опасения, что в него будут внесены случайные или намеренные изменения (система обработки изображений этого не позволит).

Способы сокрытия и применение сокрытой информации для идентификации и управления в сетях беспроводной связи описываются в патенте “Беспроводная телефония со стеганографией” [37].

Приложение Б. Исходные тексты программного комплекса информационной защиты мультимедиа-информации с использованием цифровых водяных знаков

```
//-----  
#ifndef MainH  
#define MainH  
  
#include <Classes.hpp>  
#include <Controls.hpp>  
#include <StdCtrls.hpp>  
#include <Forms.hpp>  
#include <Dialogs.hpp>  
#include <Menus.hpp>  
#include <ExtCtrls.hpp>  
#include <Graphics.hpp>  
#include <ComCtrls.hpp>  
#include <Buttons.hpp>  
  
class TfrmMain : public TForm  
{  
    __published: // IDE-managed Components  
        TButton *btnGetMsg;  
        TOpenDialog *dlgOpen;  
        TSaveDialog *dlgSave;  
        TPaintBox *pbxCvr;  
        TButton *btnPutMsg;  
        TButton *btnCvrSize;  
        TButton *btnSetCvr;  
        TEdit *edtPwd;  
        TMemo *memMsg;  
        TButton *btnClrMsg;  
        TLabel *lblPwd;  
        TLabel *lblMsg;  
        TBevel *bvlMenu;  
        TBevel *bvlCvr;  
        TImage *imgLogo;  
        TTrackBar *jpegBar;
```

```

TTrackBar *stegoBar;
 TLabel *Label1;
 TLabel *Label2;
 TStatusBar *mainBar;
 TBevel *Bevel1;
 TBitBtn *BitBtn1;
 TOpenDialog *OpenDialog1;

 void __fastcall btnGetMsgClick(TObject *Sender);
 void __fastcall btnPutMsgClick(TObject *Sender);
 void __fastcall pbxCvrPaint(TObject *Sender);
 void __fastcall btnClrMsgClick(TObject *Sender);
 void __fastcall memMsgChange(TObject *Sender);
 void __fastcall btnSetCvrClick(TObject *Sender);
 void __fastcall edtPwdChange(TObject *Sender);
 void __fastcall FormCreate(TObject *Sender);
 void __fastcall btnCvrSizeClick(TObject *Sender);
 void __fastcall BitBtn1Click(TObject *Sender);

private: // User declarations
public: // User declarations
    __fastcall TfrmMain(TComponent* Owner);
};

extern PACKAGE TfrmMain *frmMain;
#endif
//-----

//-----
#include <vcl.h>
#pragma hdrstop
USEFORM("Main.cpp", frmMain);
USEFORM("Status.cpp", frmStatus);
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    try
    {
        Application->Initialize();

```

```

        Application->Title = "Программный комплекс информационной защиты
мультимедиа-информации с использованием цифровых водяных знаков";
        Application->CreateForm(__classid(TfrmMain), &frmMain);
        Application->CreateForm(__classid(TfrmStatus), &frmStatus);
        Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    return 0;
}
//-----

//-----

#ifndef StatusH
#define StatusH
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ComCtrls.hpp>
#include <ExtCtrls.hpp>
class TfrmStatus : public TForm
{
public:           // IDE-managed Components
    TPanel *Panell;
    TProgressBar *pbrMain;
private:          // User declarations
public:           // User declarations
    __fastcall TfrmStatus(TComponent* Owner);
};
extern PACKAGE TfrmStatus *frmStatus;
#endif
//-----

```

```
-----  
#include <vcl.h>  
#pragma hdrstop  
#include "Status.h"  
#pragma package(smart_init)  
#pragma resource "*.*.dfm"  
TfrmStatus *frmStatus;  
__fastcall TfrmStatus::TfrmStatus(TComponent* Owner)  
    : TForm(Owner)  
{  
}  
-----
```

```
-----  
#include <vcl.h>  
#pragma hdrstop  
#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
#include <time.h>  
#ifdef WIN32  
    #define _WINSOCKAPI_ /* Prevent inclusion of winsock.h in windows.h */  
#/br/>  
    #include <windows.h>  
#else /* end of WIN32 */  
    #include <sys/types.h>  
    #include <sys/stat.h>  
    #include <dlfcn.h>  
#endif  
#include "stego_func.h"  
#include "rc4.h"  
#include "jpeglib.h"  
#include "unidef.h"  
#include "Main.h"  
#include "Status.h"
```

```

typedef char CIPHER_KEY;

#define PASS_0      0x00000000
#define PASS_1      0x00000000
#define PASS_2      0x00000000
#define PASS_3      0x00000000
#define PASS_4      0x00000000
#define PASS_5      0x00000000

#define MIN_PASSWORD_LEN    3

// sva_stego
out_setup_t out_setup = NULL;
in_finish_t in_finish = NULL;
out_finish_t out_finish = NULL;
exject_t exject = NULL;
inject_t inject = NULL;
in_setup_t in_setup = NULL;
check_setup_t check_setup = NULL;
check_result_t check_result = NULL;
check_finish_t check_finish = NULL;
// sva_jpeg
#define jpeg_create_compress_v(cinfo) \
    jpeg_create_compress_b((cinfo), JPEG_LIB_VERSION, \
                           (size_t) sizeof(struct jpeg_compress_struct))
#define jpeg_create_decompress_v(cinfo) \
    jpeg_create_decompress_b((cinfo), JPEG_LIB_VERSION, \
                           (size_t) sizeof(struct jpeg_decompress_struct))

typedef struct jpeg_error_mgr *(*jpeg_std_error_t)(jpeg_error_mgr * err);
typedef void (*jpeg_createcompress_t)(j_compress_ptr cinfo, int version,
size_t structsize);
typedef void (*jpeg_createdecompress_t)(j_decompress_ptr cinfo, int version,
size_t structsize);
typedef void (*jpeg_stdio_dest_t)(j_compress_ptr cinfo, FILE *outfile);
typedef void (*jpeg_stdio_src_t)(j_decompress_ptr cinfo, FILE *infile);
typedef boolean (*jpeg_start_decompress_t)(j_decompress_ptr cinfo);
typedef JDIMENSION (*jpeg_read_scanlines_t)(j_decompress_ptr cinfo, JSAMPARRAY
scanlines,JDIMENSION max_lines);
typedef bool (*jpeg_finish_decompress_t)(j_decompress_ptr cinfo);
typedef void (*jpeg_set_quality_t)(j_compress_ptr cinfo, int quality,bool
force_baseline);
typedef void (*jpeg_set_defaults_t)(j_compress_ptr cinfo);

```

```

typedef      void      (*jpeg_start_compress_t)(j_compress_ptr      cinfo,bool
write_all_tables);

typedef JDIMENSION (*jpeg_write_scanlines_t)(j_compress_ptr cinfo,JSAMPARRAY
scanlines,JDIMENSION num_lines);

typedef void (*jpeg_finish_compress_t)(j_compress_ptr cinfo);

typedef void (*jpeg_destroy_compress_t) (j_compress_ptr cinfo);

typedef void (*jpeg_destroy_decompress_t)(j_decompress_ptr cinfo);

typedef int   (*jpeg_read_header_t)(j_decompress_ptr      cinfo,      boolean
require_image);

typedef void (*set_inject_t) (void *in, void *ex);

typedef FILE * (*my_file_open_t) (const char *kont_file);

typedef void  (*my_file_close_t) ();

typedef FILE * (*my_file_openw_t) (const char *kont_file);

typedef void  (*my_file_closew_t) ();

jpeg_std_error_t jpeg_std_error_v = NULL;

jpeg_createcompress_t jpeg_create_compress_b = NULL;

jpeg_createdecompress_t jpeg_create_decompress_b = NULL;

jpeg_stdio_dest_t jpeg_stdio_dest_v = NULL;

jpeg_stdio_src_t jpeg_stdio_src_v = NULL;

jpeg_start_decompress_t jpeg_start_decompress_v = NULL;

jpeg_read_scanlines_t jpeg_read_scanlines_v = NULL;

jpeg_finish_decompress_t jpeg_finish_decompress_v = NULL;

jpeg_set_quality_t jpeg_set_quality_v = NULL;

jpeg_set_defaults_t jpeg_set_defaults_v = NULL;

jpeg_start_compress_t jpeg_start_compress_v = NULL;

jpeg_write_scanlines_t jpeg_write_scanlines_v = NULL;

jpeg_finish_compress_t jpeg_finish_compress_v = NULL;

jpeg_destroy_compress_t jpeg_destroy_compress_v = NULL;

jpeg_destroy_decompress_t jpeg_destroy_decompress_v = NULL;

jpeg_read_header_t jpeg_read_header_v = NULL;

set_inject_t set_inject_v = NULL;

my_file_open_t  my_file_open_v = NULL;

my_file_close_t my_file_close_v = NULL;

my_file_openw_t  my_file_openw_v = NULL;

my_file_closew_t my_file_closew_v = NULL;

// sva_crypto

typedef CIPHER_KEY *(*shedulekey_t)(char *passphrase, int pass_sz);

typedef int  (*encrypt_t)(CIPHER_KEY *key, char *data, int data_sz);

typedef int  (*decrypt_t)(CIPHER_KEY *key, char *data, int data_sz);

```

```

typedef int (*disposekey_t)(CIPHER_KEY *key);
typedef CIPHER_KEY *(*duplicatekey_t)(CIPHER_KEY *key);
shedulekey_t shedulekey_v = NULL;
encrypt_t encrypt_v = NULL;
decrypt_t decrypt_v = NULL;
disposekey_t disposekey_v = NULL;
duplicatekey_t duplicatekey_v = NULL;
// sva_compress
typedef int (*compress_t)(char *data, int data_sz, char **cdata, int
*cdata_sz);
typedef int (*decompress_t)(char *data, int data_sz, char **dcdata, int
*dcdata_sz);
compress_t compress_v = NULL;
decompress_t decompress_v = NULL;
// sva_ecc
typedef int (*encode_t) (char *data, int data_sz, char **cdata, int
*cdata_sz);
typedef int (*decode_t)(char *data, int data_sz, char **dcdata, int
*dcdata_sz);
encode_t encode_v = NULL;
decode_t decode_v = NULL;
// sva_natural
typedef double* (*check_natural_t) (char *filename, int filename_sz);
check_natural_t check_natural_v = NULL;
// sva_jpgtest
typedef double* (*jpg_test_t)(char *filename, int filename_sz);
jpg_test_t jpg_test_v = NULL;
int TIMEOUT = 0;
#define DEFEND 1
#pragma package(smart_init)
#pragma resource ".*.dfm"
TfrmMain *frmMain;
Graphics::TBitmap *pBitmap1 = new Graphics::TBitmap();
Graphics::TBitmap *pBitmap2 = new Graphics::TBitmap();
char kont_file[1024];
unsigned int cvrA, cvrB, cvrC, cvrD;
#ifndef WIN32
void *sys_dl_open(char *filename) { return LoadLibrary(filename); }

```

```

void *sys_dl_func(void *handle, char *name)      {                               return
GetProcAddress((HMODULE)handle, name);}
int   sys_dl_close(void *handle)                 {                               return
FreeLibrary((HMODULE)handle);                  }
#else
void *sys_dl_open(char *filename)                {   return  dlopen(filename,
RTLD_NOW);}
void *sys_dl_func(void *handle, char *name)      {   return dlsym(handle, name);
}
int   sys_dl_close(void *handle)                 {   return dlclose(handle);    }
#endif
void *crypto_hdl;
void *compress_hdl;
void *ecc_hdl;
void *natural_hdl;
void *jpgtest_hdl;
void *stego_hdl;
void *jpeg_hdl;
void free_plugin() {
    sys_dl_close(crypto_hdl);
    sys_dl_close(compress_hdl);
    sys_dl_close(ecc_hdl);
    sys_dl_close(natural_hdl);
    sys_dl_close(jpgtest_hdl);
    sys_dl_close(stego_hdl);
    sys_dl_close(jpeg_hdl);
}
int load_plugins() {
    void *inst;
    inst = sys_dl_open("sva_crypto.dll");
    if (inst) {
        shedulekey_v = (shedulekey_t) sys_dl_func(inst, "_shedulekey");
        encrypt_v = (encrypt_t) sys_dl_func(inst, "_encrypt");
        decrypt_v = (decrypt_t) sys_dl_func(inst, "_decrypt");
        disposekey_v = (disposekey_t) sys_dl_func(inst, "_disposekey");
        duplicatekey_v      =      (duplicatekey_t)      sys_dl_func(inst,
"_duplicatekey");
        crypto_hdl = inst;
    }
}

```

```

inst = sys_dl_open("sva_compress.dll");
if (inst) {
    compress_v = (compress_t) sys_dl_func(inst, "_compress");
    decompress_v = (decompress_t) sys_dl_func(inst, "_decompress");
    compress_hdl = inst;
}
inst = sys_dl_open("sva_ecc.dll");
if (inst) {
    encode_v = (encode_t) sys_dl_func(inst, "_encode");
    decode_v = (decode_t) sys_dl_func(inst, "_decode");
    ecc_hdl = inst;
}
inst = sys_dl_open("sva_natural.dll");
if (inst) {
    check_natural_v      =      (check_natural_t)      sys_dl_func(inst,
"_check_natural");
    natural_hdl = inst;
}
inst = sys_dl_open("sva_jpgtest.dll");
if (inst) {
    jpg_test_v = (jpg_test_t) sys_dl_func(inst, "_jpg_test");
    jpgtest_hdl = inst;
}
inst = sys_dl_open("sva_stego.dll");
if (inst) {
    out_setup = (out_setup_t) sys_dl_func(inst, "_out_setup");
    in_finish = (in_finish_t) sys_dl_func(inst, "_in_finish");
    out_finish = (out_finish_t) sys_dl_func(inst, "_out_finish");
    exject = (exject_t) sys_dl_func(inst, "_exject");
    inject = (inject_t) sys_dl_func(inst, "_inject");
    in_setup = (in_setup_t) sys_dl_func(inst, "_in_setup");
    check_setup = (check_setup_t) sys_dl_func(inst, "_check_setup");
    check_result      =      (check_result_t)      sys_dl_func(inst,
"_check_result");
    check_finish      =      (check_finish_t)      sys_dl_func(inst,
"_check_finish");
    stego_hdl = inst;
}
inst = sys_dl_open("sva_jpeg.dll");

```

```

if (inst) {
    TIMEOUT = 1;
    jpeg_std_error_v      =      (jpeg_std_error_t)      sys_dl_func(inst,
"_jStdError");
    jpeg_create_compress_b = (jpeg_createcompress_t)  sys_dl_func(inst,
"_jCreaCompress");
    jpeg_create_decompress_b =          (jpeg_createdecompress_t)
sys_dl_func(inst, "_jCreaDecompress");
    jpeg_stdio_dest_v     =      (jpeg_stdio_dest_t)     sys_dl_func(inst,
"_jStdDest");
    jpeg_stdio_src_v      =      (jpeg_stdio_src_t)      sys_dl_func(inst,
"_jStdSrc");
    jpeg_start_decompress_v =          (jpeg_start_decompress_t)
sys_dl_func(inst, "_jStrtDecompress");
    jpeg_read_scanlines_v = (jpeg_read_scanlines_t)  sys_dl_func(inst,
"_jReadScanlines");
    jpeg_finish_decompress_v = (jpeg_finish_decompress_t)
sys_dl_func(inst, "_jFinDecompress");
    jpeg_set_quality_v    =      (jpeg_set_quality_t)    sys_dl_func(inst,
"_jSetQuality");
    jpeg_set_defaults_v   =      (jpeg_set_defaults_t)   sys_dl_func(inst,
"_jSetDefaults");
    jpeg_start_compress_v = (jpeg_start_compress_t)  sys_dl_func(inst,
"_jStrtCompress");
    jpeg_write_scanlines_v =          (jpeg_write_scanlines_t)
sys_dl_func(inst, "_jWrtScanlines");
    jpeg_finish_compress_v = (jpeg_finish_compress_t)
sys_dl_func(inst, "_jFinCompress");
    jpeg_destroy_compress_v = (jpeg_destroy_compress_t)
sys_dl_func(inst, "_jDestCompress");
    jpeg_destroy_decompress_v = (jpeg_destroy_decompress_t)
sys_dl_func(inst, "_jDestDecompress");
    jpeg_read_header_v    =      (jpeg_read_header_t)    sys_dl_func(inst,
"_jReadHeader");
    set_inexject_v        =      (set_inexject_t)        sys_dl_func(inst,
"_set_inexject");
    my_file_open_v         =      (my_file_open_t)       sys_dl_func(inst,
"_my_file_open");

```

```

        my_file_close_v      =      (my_file_close_t)      sys_dl_func(inst,
"__my_file_close");
        my_file_openw_v       =      (my_file_openw_t)      sys_dl_func(inst,
"__my_file_openw");
        my_file_closew_v     =      (my_file_closew_t)    sys_dl_func(inst,
"__my_file_closew");
        jpeg_hdl = inst;
        set_inexject_v(inject, exject);
    }
    return 1;
}

_fastcall TfrmMain::TfrmMain(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TfrmMain::btnGetMsgClick(TObject *Sender)
{
    struct jpeg_decompress_struct cinfo;
    struct jpeg_error_mgr jerr;
    FILE * infile;
    JSAMPARRAY buffer;
    int row_stride;
    int mypl=stegoBar->Position;
    if (mypl<1) mypl=1;
    if (mypl>8) mypl=8;
    out_setup("1.1",           edtPwd->Text.c_str(),          strlen(edtPwd-
>Text.c_str()),mypl);
    if ((infile = my_file_open_v(kont_file)) == NULL) {
        fprintf(stderr, "can't open %s\n", kont_file);
        return;
    }
    frmStatus->pbrMain->Position = 0;
    frmStatus->Visible=true;
    cinfo.err = jpeg_std_error_v(&jerr);
    jpeg_create_decompress_v(&cinfo);
    jpeg_stdio_src_v(&cinfo, infile);
    jpeg_read_header_v(&cinfo, TRUE);
    jpeg_start_decompress_v(&cinfo);
    row_stride = cinfo.output_width * cinfo.output_components;
}

```

```

        buffer      =      (*cinfo.mem->alloc_sarray)    ((j_common_ptr)      &cinfo,
JPOOL_IMAGE, row_stride, 1);
        pBitmap1->Height = cinfo.output_height;
        pBitmap1->Width = cinfo.output_width;
        frmMain->Refresh();
        while (cinfo.output_scanline < cinfo.output_height) {
            frmStatus->pbrMain-
>Position=100*(cinfo.output_scanline+1)/cinfo.output_height;
            frmStatus->Refresh();
            jpeg_read_scanlines_v(&cinfo, buffer, 1);
        }
        jpeg_finish_decompress_v(&cinfo);
        my_file_close_v();
        out_finish();
        CIPHER_KEY * mykey1;
        FILE *infile1;
        FILE *outfile1;
        char mydata[8192];
        int mydatasize;
        if ((infile1=fopen("1.1","rb"))!=NULL) {
            mykey1=shedulekey_v(edtPwd->Text.c_str(),strlen(edtPwd->Text.c_str()));
            outfile1=fopen("msg.dat", "wb");
            mydatasize=fread(mydata,1,4096,infile1);
            while(mydatasize) {
                decrypt_v(mykey1,mydata,mydatasize);
                fwrite(mydata,1,mydatasize,outfile1);
                mydatasize=fread(mydata,1,4096,infile1);
            }
            fclose(outfile1);
            fclose(infile1);
            disposekey_v(mykey1);
        }
        memMsg->Lines->LoadFromFile("msg.dat");
        frmStatus->Visible = false;
    }
void __fastcall TfrmMain::pbxCvrPaint(TObject *Sender)
{
    pbxCvr->Canvas->Draw(0, 0, pBitmap2);
}

```

```

void __fastcall TfrmMain::btnClrMsgClick(TObject *Sender)
{
    memMsg->Clear();
    btnPutMsg->Enabled = false;
}

void __fastcall TfrmMain::memMsgChange(TObject *Sender)
{
//    if (!memMsg->Lines->Count)
//        btnPutMsg->Enabled = false;
//    else
//        if (strlen(edtPwd->Text.c_str()) > MIN_PASSWORD_LEN-1)
//            btnPutMsg->Enabled = true;
}

void __fastcall TfrmMain::edtPwdChange(TObject *Sender)
{
    if (strlen(edtPwd->Text.c_str()) < MIN_PASSWORD_LEN) {
        btnGetMsg->Enabled = false;
        btnCvrSize->Enabled = false;
        btnPutMsg->Enabled = false;
    }
    else {
        if (strlen(kont_file) > 0) {
            btnGetMsg->Enabled = true;
            btnCvrSize->Enabled = true;
            btnPutMsg->Enabled = true;
        }
    }
}

void __fastcall TfrmMain::FormCreate(TObject *Sender)
{
    uint32_t pass[6];
    RC4_KEY rc4;
    TColor buf;
    pass[1] = PASS_1;
    pass[0] = PASS_0;
    pass[3] = PASS_3;
    pass[5] = PASS_5;
    pass[4] = PASS_4;
    pass[2] = PASS_2;
}

```

```

setup_rc4(&rc4, (uint8_t *) &pass, 13);
for (int x = 0; x < imgLogo->Picture->Bitmap->Width; x++) {
    for (int y = 0; y < imgLogo->Picture->Bitmap->Height; y++) {
        buf = imgLogo->Picture->Bitmap->Canvas->Pixels[x][y];
        encode_rc4(&rc4, (uint8_t *) &buf, 3);
        imgLogo->Picture->Bitmap->Canvas->Pixels[x][y] = buf;
    }
}
load_plugins();
if ((jpeg_hdl==NULL) || (stego_hdl==NULL)) {
Application->MessageBox("Ошибка при подключении обязательных модулей
\n","Критическая ошибка",MB_ICONERROR);
Application->Terminate();
}
}
void __fastcall TfrmMain::btnCvrSizeClick(TObject *Sender)
{
    char *fileoutname;
    struct jpeg_decompress_struct cinfo;
    struct jpeg_compress_struct coutfo;
    struct jpeg_error_mgr jerr,jerrout;
    FILE *infile;
    FILE *outfile;
    JSAMPARRAY buffer;
    int row_stride;
    memMsg->Lines->SaveToFile("1.2");
// if (dlgSave->Execute()) {
    int mypl=stegoBar->Position;
    if (mypl<1) mypl=1;
    if (mypl>8) mypl=8;
    check_setup("",      edtPwd->Text.c_str(),      strlen(edtPwd-
>Text.c_str()),mypl);
        frmStatus->pbrMain->Position = 0;
    frmStatus->Visible = true;
    if ((infile = my_file_open_v(kont_file)) == NULL) {
        fprintf(stderr, "can't open %s\n", kont_file);
        return;
    }
    cinfo.err = jpeg_std_error_v(&jerr);
}

```

```

jpeg_create_decompress_v(&cinfo);
jpeg_stdio_src_v(&cinfo, infile);
jpeg_read_header_v(&cinfo, TRUE);
jpeg_start_decompress_v(&cinfo);
coutfo.err = jpeg_std_error_v(&jerrout);
jpeg_create_compress_v(&coutfo);

fileoutname = (ExtractFilePath(Forms::Application->ExeName)+ "tmp.tmp").c_str();

if ((outfile = my_file_openw_v(fileoutname)) == NULL) {
    fprintf(stderr, "can't open %s\n", fileoutname);
    return;
}

jpeg_stdio_dest_v(&coutfo, outfile);
coutfo.image_width = cinfo.output_width;
coutfo.image_height = cinfo.output_height;
coutfo.input_components = 3;
coutfo.in_color_space = JCS_RGB;
jpeg_set_defaults_v(&coutfo);
int myq=jpegBar->Position;
if (myq<1) myq=1;
if (myq>100) myq=100;
jpeg_set_quality_v(&coutfo, myq, TRUE);
jpeg_start_compress_v(&coutfo, TRUE);
row_stride = cinfo.output_width * cinfo.output_components;
buffer = (*cinfo.mem->alloc_sarray) ((j_common_ptr) &cinfo,
JPOOL_IMAGE, row_stride, 1);
frmMain->Refresh();
while (cinfo.output_scanline < cinfo.output_height) {
    frmStatus->pbrMain->Position=100*(cinfo.output_scanline+1)/cinfo.output_height;
    frmStatus->Refresh();
    jpeg_read_scanlines_v(&cinfo, buffer, 1);
    jpeg_write_scanlines_v(&coutfo, buffer, 1);
}
jpeg_finish_decompress_v(&cinfo);
jpeg_destroy_decompress_v(&cinfo);
my_file_close_v();
jpeg_finish_compress_v(&coutfo);
my_file_closew_v();

```

```

jpeg_destroy_compress_v(&coutfo);
CHECKDATA *chkdta;
chkdta =check_result();
int ksize, fsize;
double pr,fpr;
ksize=(chkdta->u0+chkdta->u1)/8-4;
pr=(1-abs((double)chkdta->a0-(double)chkdta->a1)/((double)chkdta-
>a0+(double)chkdta->a1))*100;
check_finish();
fileoutname = ExtractFilePath(Forms::Application-
>ExeName)+"tmp.tmp\0").c_str();
if ((infile = fopen(fileoutname, "rb")) == NULL) {
    fprintf(stderr, "can't open %s\n", kont_file);
    return;
}
fseek(infile, 0, SEEK_END);

fsize=f.tell(infile);
fpr= 100*((double)ksize/(double)fsize);
mainBar->Panels->Items[1]->Text=" Емкость "+AnsiString(ksize)+" из "+
fsize+" (байт) "+ fpr+"%";
mainBar->Panels->Items[2]->Text=" Пригодность "+AnsiString(pr)+"%";
fclose(infile);
fileoutname = ExtractFilePath(Forms::Application-
>ExeName)+"tmp.tmp\0").c_str();
if(FileExists(fileoutname)) DeleteFile(fileoutname);
frmStatus->Visible = false;
}

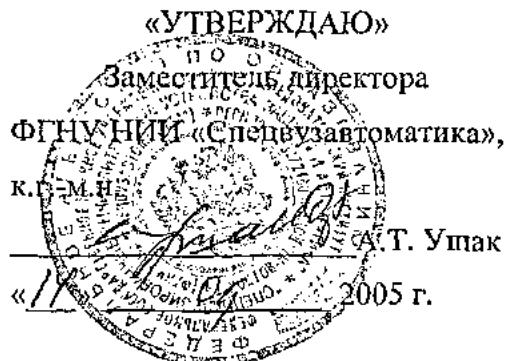
void __fastcall TfrmMain::BitBtn1Click(TObject *Sender)
{
AnsiString errStr;
errStr+="sva_jpeg.dll: ";
if (jpeg_hdl==NULL) errStr+="не подключен \n"; else errStr+="подключен \n";
errStr+="sva_stego.dll: ";
if (stego_hdl==NULL) errStr+="не подключен\n"; else errStr+="подключен \n";
errStr+="sva_keygen.dll: ";
if (stego_hdl==NULL) errStr+="не подключен\n"; else errStr+="подключен\n";
errStr+="sva_crypto.dll: ";
if (crypto_hdl==NULL) errStr+="не подключен\n"; else errStr+="подключен\n";

```

```
errStr+="sva_compress.dll: ";
if      (compress_hndl==NULL)      errStr+="не      подключен\n";      else
errStr+="подключен\n";
errStr+="sva_ecc.dll: ";
if (ecc_hndl==NULL) errStr+="не подключен\n"; else errStr+="подключен\n";
errStr+="sva_natural.dll: ";
if (natural_hndl==NULL) errStr+="не подключен\n"; else errStr+="подключен\n";
errStr+="sva_jpgtest.dll: ";
if (jpgtest_hndl==NULL) errStr+="не подключен\n"; else errStr+="подключен\n";
Application->MessageBox( ("Подключены                     следующие
модули:\n\n"+errStr+"\n").c_str(), "Информация о модулях", MB_ICONINFORMATION);
}
//-----
```

Приложение В. Акты об использовании результатов диссертационной работы

Федеральное агентство по образованию
Федеральное государственное
научное учреждение
Научно-исследовательский институт
«Специализированные
вычислительные устройства
защиты и автоматика»
344002, г. Ростов-на-Дону,
пер. Газетный, 51. Тел. 297-50-84
14.09.05 N 165
на N _____ от _____



АКТ

об использовании результатов диссертационной работы Балакина А.В.
"Разработка архитектуры программного комплекса и методов
информационной защиты мультимедиа-информации
с использованием цифровых водяных знаков"

Настоящим подтверждается, что в период с 1.02.2004 по 20.11.2004 при выполнении научно-исследовательских работ "Бородино" (инв. № 2/2004-СВА-2/К-01/2), "Ключ" (инв. № 14/2004-СВА-3/М-01/3) и опытно-конструкторской работы "Заслонка" (инв. № 7/2004-СВА-5/М-02/1) в ФГНУ НИИ "Спецвузавтоматика" использованы теоретические и практические результаты, полученные в диссертационной работе Балакина А.В. "Разработка архитектуры программного комплекса и методов информационной защиты мультимедиа-информации с использованием цифровых водяных знаков":

- а) На основе разработанного метода количественной оценки вносимых при внедрении цифровых водяных знаков искажений, Балакиным А.В. был разработан динамически загружаемый программный модуль контроля искажений (Свидетельство об официальной регистрации программы для ЭВМ №2005610537/РОСПАТЕНТ. - М., 28.02.2005.), использованный при проведении численных экспериментов в НИР "Бородино";
- б) На основе разработанных Балакиным А.В. программных интерфейсов взаимодействия была разработана система динамически

подключаемых модулей (Свидетельство об официальной регистрации программы для ЭВМ №2003611307/РОСПАТЕНТ. - М., 29.05.2003.), обеспечивающих аутентификацию сеанса связи в макете, разработанном в рамках НИР "Ключ";

в) Разработанная Балакиным А.В. архитектура программного комплекса и система интерфейсов для подключения модулей стороннего разработчика использована при проектировании и разработке программной части аппаратно-программного комплекса, созданного в рамках ОКР "Заслонка".

Использование результатов, полученных Балакиным А.В., позволило сократить время и затраты на разработку программного обеспечения, а также упростить процедуры его настройки и модернизации во время эксплуатации.

Председатель комиссии

Заведующий лабораторией,
к.т.н., с.н.с.

"14" сентября 2005 г.

Леднов Д.А.

Члены комиссии

Заведующий отделом,
к.т.н.

"14" 09 2005 г.

Берг О.Ю.

Научный сотрудник,

к.ф.-м.н.

"14" сентября 2005 г.

Репаков С.А.

УТВЕРЖДАЮ

29.07.2005 № ОИ / 815



Ю.П. Щербак

АКТ

О внедрении результатов диссертационной работы Балакина А.В.
«Разработка архитектуры программного комплекса и методов информационной
защиты мультимедиа-информации с использованием цифровых водяных
знаков» в учебный процесс в Саровском государственном физико-техническом
институте (г. Саров Нижегородской области)

Настоящим подтверждается, что теоретические и практические
результаты, полученные в диссертационной работе Балакина А.В. «Разработка
архитектуры программного комплекса и методов информационной защиты
мультимедиа-информации с использованием цифровых водяных знаков»:

- метод контроля целостности информации на основе извлеченных
данных;
- обобщенная архитектура программного комплекса контроля
целостности мультимедиа-информации, которая основана на мультиагентном
подходе;
- открытые интерфейсы для подключения программного обеспечения
стороннего разработчика, позволяющие изменять функциональность отдельных
модулей и комплекса в целом

внедрены в учебный процесс на основании решения кафедры «Радиофизика и
электроника» Саровского государственного физико-технического института
(протокол № 9 от 28.06.2005 г.).

Указанные результаты включены в дисциплины «Криптография и
специальные исследования» и «Основы криптографии» в качестве глав в
учебно-методическом пособии «Разработка архитектуры программного
комплекса информационной защиты ведомственной локальной сети и рабочих
станций».

Заведующий кафедрой
«Радиофизика и электроника»
доктор технических наук, профессор,
заслуженный деятель науки

Мартынов Александр
Петрович (83130)-44867
МА 2 28.07.2005

 — А.И. Астайкин

УТВЕРЖДАЮ
начальника ГНИИ ПТЗИ
ФСТЭК России

кандидата технических наук, доцент

А.А.Бурушкин



АКТ

об использовании результатов диссертационной работы Балакина А.В.
"Разработка архитектуры программного комплекса и методов информационной защиты
мультимедиа-информации с использованием цифровых водяных знаков"

Комиссия ГНИИ ПТЗИ ФСТЭК России в составе: председателя – начальника отдела кандидата технических наук, старшего научного сотрудника Язова Ю.К. и членов комиссии: заместителя начальника отдела кандидата технических наук, старшего научного сотрудника Паринова И.В., старшего научного сотрудника Дмитриева А.С., рассмотрела состояние использования результатов диссертационной работы Балакина А.В. и установила следующее.

1. Автором разработаны, основанные на использовании псевдометрик, методы, алгоритмы количественной оценки вносимых искажений при внедрении цифровых водяных знаков и метод выделения пространства сокрытия из различных форматов хранения мультимедийной информации, основанный на модели естественности мультимедийной информации (графических и аудио-файлов).

2. Содержащиеся в диссертационной работе результаты были использованы ГНИИ ПТЗИ ФСТЭК России при разработке макетов программных средств обнаружения и противодействия несанкционированной скрытной передаче информации с использованием графических и аудио-файлов, а также средств защиты информации в информационно-телекоммуникационных сетях общего пользования, сетях органов государственного управления на основе ее стеганографического преобразования, разработанных в рамках НИР "Картина-А".

Председатель комиссии:
начальник отдела
кандидат технических наук,
старший научный сотрудник

Язов Ю.К.

члены комиссии:
заместитель начальника отдела
кандидат технических наук,
старший научный сотрудник

старший научный сотрудник

Паринов И.В.

Дмитриев А.С.

УТВЕРЖДАЮ
Профессор по научной работе
Московского технического
университета связи и информатики

" 3 "

Ф.И.О. Аттестующего

В.С.Алешин

2005 г.

АКТ

об использовании результатов докторской работы Балакина А.В.
"Разработка архитектуры программного комплекса и методов информационной
защиты мультимедиа-информации с использованием цифровых водяных
знаков"

Научно-техническая комиссия в составе: начальника отдела Аджемова С.С., заведующего лабораторией Романова Э.Ю. и инженера Пчелки М.В. составила настоящий акт в том, что разработанные в докторской работе Балакина А.В. методы внедрения и извлечения цифровых водяных знаков из мультимедиа-контейнеров, основанные на модели естественности мультимедиа-информации и метод контроля целостности мультимедиа-контейнеров на основе извлеченных цифровых водяных знаков использованы в Московском Техническом Университете Связи и Информатики в период с 08.04.2002г. по 30.11.2004г. в ходе выполнения научно-исследовательской работы "Полтава-ТС", выполненной по государственному контракту №1/02-Э от 9.04.02г.

Данная работа имеет большой научно-практический интерес и позволяет осуществлять эффективное проектирование стойких систем внедрения цифровых водяных знаков. Предложенные в докторской интерфейсы взаимодействия управляющих и функциональных компонент использованы при разработке специальных аппаратно-программных средств.

Члены комиссии

Начальник НИО

С.С.Аджемов

Заведующий НИЛ

Э.Ю.Романов

Инженер

М.В.Пчелка