Санкт-Петербургский государственный университет

Дамбаева Алина Юрьевна

Выпускная квалификационная работа бакалавра

Программная реализация технологии хранения и передачи персональных данных

Направление 02.03.02

«Фундаментальная информатика и информационные технологии»

ООП СВ.5003.2018 «Программирование и информационные технологии»

Научный руководитель:

доктор технических наук, профессор

кафедры компьютерного моделирования

и многопроцессорных систем,

Щеголева Надежда Львовна

Санкт-Петербург

2022

Содержание

Введение

В настоящее время одной из наиболее опасных угроз в информационном пространстве является утечка данных, то есть неправомерная передача конфиденциальных данных лицам, которые не уполномочены использовать эти данные. По данным российского сервиса разведки утечек данных Data Leakage & Breach Intelligence (DLBI) медицинские учреждения входят в число организаций, в которых чаще всего фиксируются утечки данных [1].

Одни из наиболее известных случаев утечек медицинских данных за последнее время связаны с пандемией новой коронавирусной инфекции COVID-19. В связи со сложившейся неблагоприятной эпидемиологической обстановкой проводилось много медицинских исследований, например, анализ полимеразной цепной реакции (ПЦР), с целью выявления заболевания у людей. Помимо огромного массива данных о результатах исследований и о заболевших необходимо было собирать и хранить информацию о лицах, контактировавших с заболевшими, и о лицах, прошедших вакцинацию.

9 декабря 2020 года в открытом доступе появились данные 300 000 человек, переболевших COVID-19. Данные содержат персональную информацию: ФИО, адрес проживания и регистрации, результаты анализов. В январе 2022 появились новости о том, что в даркнете на продажу выставили данные 48 млн сертификатов о вакцинации от коронавируса за $100 тысяч. Предоставленный образец базы данных содержит следующую информацию о вакцинированных: первые буквы ФИО на русском и английском языках, дату рождения, УНРЗ (уникальный номер регистровой записи пациента), первые две цифры серии и последние три цифры номера паспорта, название вакцины на русском и английском языках, QR-код в формате PNG, закодированный в Base64, и срок его действия.

Утечки данных могут привести к серьезным последствиям как для пострадавших лиц, так и для организаций, допустивших такой инцидент. Размещение в открытом доступе персональных данных и информации о течении болезни является не только нарушением законодательства о

персональных данных, но и нарушает неприкосновенность частной жизни и врачебной тайны. В такой ситуации пострадавшее лицо может подать в суд, чтобы привлечь к ответственности виновную организацию. Люди, чьи данные неправомерно были переданы третьим лицам, могут стать жертвами злоумышленников.

Многие люди опасаются утечки сведений об их медицинских исследованиях из-за возможных последствий: помимо перечисленных выше, есть риски возникновения сложностей на работе или в учебных заведениях при выявлении определенных заболеваний. Иногда это приводит к тому, что люди избегают получения медицинской помощи из-за этого риска. Особенно часто люди могут избегать получения психиатрической помощи из-за существующей стигматизации психически больных людей, которая значительно снижает уровень жизни этих лиц. Люди с психическими расстройствами подвержены дискриминации в семье, на работе, в личной жизни и общественной деятельности [2].

Возможным способом предотвращения негативных последствий утечек сведений является их обезличивание. Но в настоящее время анонимное оказание медицинской помощи в России осложнено рядом причин, даже при платном оказании медицинских услуг. В некоторых случаях законодательство Российской Федерации предусматривает платную медицинскую помощь на анонимной основе. Однако платные медицинские услуги требуют оформления договора, но для заключения договора на анонимной основе отсутствует какое-либо специальное законодательное или иное нормативное урегулирование. В связи с этим возникают бюрократические сложности с получением медицинской помощи на анонимной основе.

Одной из причин утечки данных является человеческий фактор: нередко это происходит по вине персонала, при этом сотрудники могут действовать как умышленно, так и случайно допустить распространение конфиденциальной информации [3]. В том числе утечка сведений о

переболевших COVID-19 в декабре 2020 года произошла по вине сотрудников, а не по причине взломов или несанкционированного доступа.

В связи со всем вышеперечисленным важно обеспечивать безопасное хранение и передачу данных, чтобы предотвращать негативные следствия возможной утечки конфиденциальных данных. Одним из возможных способов обеспечения такой безопасности является шифрование данных, чтобы даже в случае передачи файлов с данными третьим лицам не было возможности воспользоваться их содержимым. И так как распространение информации именно о психических заболеваниях может иметь особенно разрушительные последствия для людей, в национальном медицинском исследовательском центре психиатрии и неврологии им. В. М. Бехтерева возникла необходимость обеспечить надежное хранение сведений о пациентах. Данная работа посвящена решению именно этой задачи.

Постановка задачи

Целью данной работы является программная реализация эффективного метода для безопасного хранения и передачи медицинских данных для последующего его применения в НМИЦ ПН им. В. М. Бехтерева.

Первой задачей исследования является определение подходящего метода, который обеспечил бы шифрование данных таким образом, чтобы было неизвестно, что в носителе информации содержатся какие-то конфиденциальные сведения. Следующей задачей является проектирование и реализация приложения, содержащего найденный метод, анализ его работы, повышение эффективности.

Обзор литературы

Персональные данные – любая информация, относящаяся к прямо или косвенно определенному или определяемому физическому лицу (субъекту персональных данных), как определено в п. 1 ст. 3 Федерального закона от 27 июля 2006 г. №152-ФЗ "О персональных данных" [4]. Из этого определения следует, что к персональным данным можно отнести почти любую информацию о человеке: ФИО, пол, возраст, место проживания, образование, семейное положение, паспортные данные и др. В том числе к персональным данным относятся и сведения о здоровье человека.

Защита персональных данных – это комплекс организационных и технических мер, исключающих доступ к информации, содержащей персональные данные, лиц, не обладающих соответствующими полномочиями, и исключающих возможность неправомерного использования такой информации [5].

Для защиты данных необходимо обеспечить их безопасное хранение и передачу. Как утверждается в работе [6], в настоящее время в медицинских учреждениях России наиболее часто используют следующие способы хранения данных: реляционные системы управления базами данных (СУБД) и нереляционные СУБД, в частности, документоориентированные СУБД, а также часто медицинские информационные системы (МИС) основываются на совмещении этих подходов, то есть на смешанном варианте, так как в чистом виде перечисленные подходы могут быть недостаточными для представления медицинских данных.

Основные стандарты для хранения медицинских данных, регулирующие основные принципы того, как именно данные хранятся, были описаны различными международными организациями. Проектирование новых МИС в России происходит с учетом этих стандартов. Примеры стандартов: HL7 (США), ASTM TC E31 (США), CEN TC 251 (Европа), openEHR (Великобритания, Австралия). Ключевые из них – HL7 и openEHR. В Таблица 1 приведен их сравнительный анализ.

Таблица 1: Сравнение стандартов HL7 и openEHR

Критерий	HL7	openEHR
Уровень детализации описания документа	Три варианта глубины детализации	Произвольная детализация
Гибкость структуры данных	Большие возможности по созданию структуры, но существует сложность в жесткой привязке к стандартным классификаторам	Возможность формировать произвольные структуры представления данных
Технология обмена данными	Существует технология передачи сообщений по стандартам западных информационных систем	Есть возможность интеграции с внешними системами
Безопасность персональных данных	Стандарт предусматривает возможность ролевой модели доступа к данным	Стандарт предусматривает принцип разделения персональных и медицинских данных, учет версий документов, учет прав доступа и возможность использования электронной цифровой подписи

В результате сравнения можно прийти к выводу, что openEHR является наиболее подходящим стандартом при проектировании МИС ввиду своей гибкости при выполнении различных задач [6].

Однако исследования показывают, что во многих учреждениях используется не тот способ представления медицинских данных, который был бы наиболее подходящим для выполнения требуемых задач, а тот способ, который был выбран на начальном этапе развития данной МИС. Все последующие прикладные решения зависят от возможностей изначально выбранной технологической платформы [6]. По этой причине для НМИЦ ПН

им. В.М. Бехтерева требуется разработать такое решение для защиты медицинских данных, которое можно было бы внедрить в существующую систему и которое повышало бы безопасность хранения и передачи этих данных.

В работе [7] утверждается, что в последнее время одним из наиболее эффективных подходов является defense-in-depth, или многослойная, эшелонированная защита данных, то есть использование нескольких последовательных мер для предотвращения доступа к информации третьих лиц. В рамках данного подхода используется несколько уровней защиты данных, что повышает безопасность системы и снижает риски несанкционированного доступа к сведениям.

Один из уровней защиты может быть реализован с помощью алгоритма криптографического шифрования, который обеспечивает защиту данных путем преобразования в непонятный и бесполезный для злоумышленника вид. Алгоритмы шифрования бывают симметричными, то есть с использованием одного ключа, и асимметричными, то есть с использованием двух ключей. В работе [8] проведено сравнение разных алгоритмов и утверждается, что работа симметричных алгоритмов намного быстрее и требует меньше вычислительной мощности, чем при асимметричном шифровании. По этой причине для текущей задачи решено использовать симметричное шифрование. В работе [9] выполнено сравнение симметричных алгоритмов шифрования Data Encryption Standard (DES) и Advanced Encryption Standard (AES) и заключено, что AES работает быстрее. В работе [10] при сравнении тех же алгоритмов утверждается, что AES менее уязвим для атак методом перебора. На основе изученных сравнений алгоритмов симметричного шифрования выбран алгоритм AES для использования в текущей работе.

Следующий уровень защиты может быть обеспечен с помощью методов стеганографии, то есть с помощью алгоритма, который скрывает не только содержание хранимой информации, но и сам факт хранения или передачи каких-то данных. В работе [11] рассматривается алгоритм встраивания QR-

кодов в цветные изображения в задачах биометрии. Суть алгоритма заключается в том, что данные встраиваются в наименее значимые слои изображения, поэтому внешне изменения изображения не заметны. Этот алгоритм подходит и для текущей задачи, если в QR-коде хранить медицинские данные. С помощью этого метода медицинские данные можно шифровать и хранить в изображениях таким образом, что на взгляд нельзя определить, что в изображении что-то скрыто.

Глава 1. Теоретическая часть

В этой главе будут рассмотрены теоретические основы для проектирования приложения. Будут описаны алгоритмы, использование которых является основой для программной реализации решения поставленной задачи.

1.1 Алгоритм шифрования AES

В качестве подхода для обеспечения безопасности медицинских данных при их хранении и передаче выбрана эшелонированная защита данных, которая заключается в обеспечении нескольких последовательных уровней защиты информации. Для реализации первого уровня безопасности в этом подходе решено применять алгоритм криптографического шифрования Advanced Encryption Standard (AES).

Алгоритм AES (или Rijndael) является симметричным и блочным. Суть симметричных алгоритмов заключается в том, что один и тот же ключ используется как для шифрования, так и для дешифрования информации. Блочный шифр – тип симметричного шифра, который оперирует блоками данных. Эти блоки должны быть фиксированной длины, в случае AES размер одного блока – 16 байт (128 бит). Исходные данные большей длины разбиваются на блоки определенного размера, и если последний блок содержит меньше символов, чем необходимо, то его дополняют до нужной длины. Блоки данных представлены в виде матрицы размером 4×4.

Существует несколько разновидностей данного алгоритма: AES-128, AES-192, AES-256. Их различие заключается в длине ключа: он может быть 128, 192 или 256 бит. Для решения текущей задачи решено использовать AES-128, так как ключа размером 128 бит достаточно для шифрования информации в рамках этой задачи. При таком размере ключа количество раундов в алгоритме равно 10.

Для каждого раунда используется свой ключ. Процедура Key Expansion расширяет основной ключ шифрования, подаваемый на вход алгоритму, для генерации раундовых ключей. Полученный после расширения ключ состоит

из 44 слов длиной по 4 байт: 4 слова на основной ключ и по 4 слова на каждый из 10 этапов для раундовых ключей.

Раунды представляют собой последовательность определенных преобразований, которые будут описаны далее. Раунд 0 состоит только из функции Add Round Key. Все последующие раунды, за исключением последнего, состоят из следующих шагов: Substitute Bytes, Shift Rows, Mix Columns, Add Round Key. Финальный раунд содержит этапы Substitute Bytes, Shift Rows, Add Round Key [12]. На Рис. 1 представлен алгоритм AES как последовательность раундов.

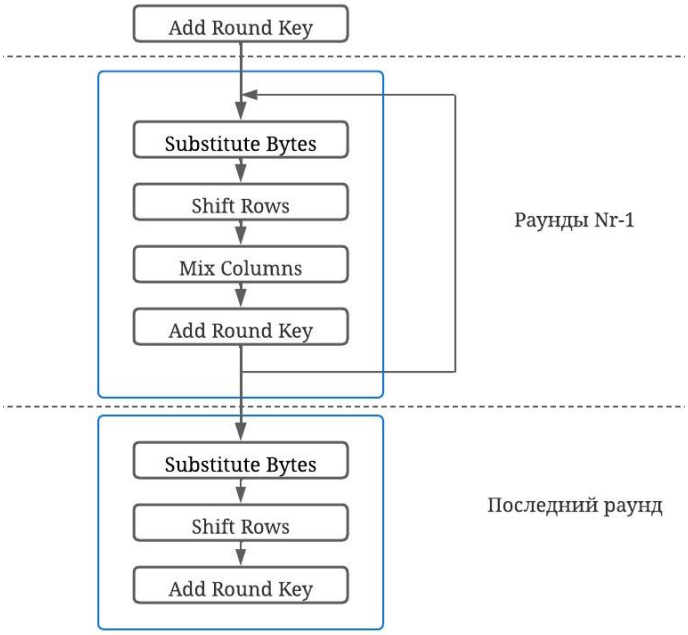

Рис. 1: Алгоритм AES

Далее описаны преобразования, выполняемые в раундах алгоритма.

- Add Round Key Transformation

Для этого преобразования применяется операция XOR к битам входящего блока и ключу раунда.

- Substitution Bytes

На данном этапе происходит преобразование на основе S-box, изображенной на Рис. 2. S-box – таблица 16×16, содержащая значения для замены исходного байта: первые 4 бита определяют строку таблицы, а

последние 4 бита – столбец. Исходный байт заменяют значением на пересечении данной строки и столбца.

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Рис. 2: S-box

- Shift Rows Transformation

На данном шаге используется следующее преобразование: первая строка остается неизменной, во второй строке происходит сдвиг на 1 позицию влево, в третьей строке – сдвиг на 2 позиции влево, в четвертой строке – на 3 позиции влево.

- Mix Columns Transformation

На этом этапе преобразование происходит по столбцам матрицы-блока. Каждый столбец умножается на матрицу на Рис. 3.

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Рис. 3: Матрица для Mix Columns Transformation

Для расшифровки данных, закодированных с помощью AES-128, нужно применить обратные преобразования в обратном порядке. Раунд 0 остается таким же и состоит из преобразования Add Round Key. Раунды, кроме последнего, содержат следующие операции: Inverse Shift Rows, Inverse Substitution Bytes, Add Round Key, Inverse Mix Columns. Финальный раунд состоит из Inverse Shift Rows, Inverse Substitution Bytes, Add Round Key.

- Inverse Substitution Bytes

Для обратного преобразования при декодировании используется Inverse S-box (Рис. 4).

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1x	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2x	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3x	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4x	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5x	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6x	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7x	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8x	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9x	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
ax	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
bx	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
cx	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
dx	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
ex	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
fx	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Рис. 4: Inverse S-box

- Inverse Shift Rows Transformation

Для обратного алгоритма используется сдвиг вправо, а не влево как в прямом преобразовании. Первая строка остается неизменной, во второй строке происходит сдвиг на 1 позицию вправо, в третьей строке – сдвиг на 2 позиции вправо, в четвертой строке – на 3 позиции вправо.

- Inverse Mix Columns Transformation

В обратном преобразовании происходит умножение столбцов на матрицу на Рис. 5, отличную от матрицы в прямом алгоритме.

$$\begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix}$$

Рис. 5: Матрица для Inverse Mix Columns Transformation

1.2 Штриховое кодирование

Для того, чтобы выполнить последующие этапы защиты данных, к ним требуется применить технологии штрихового кодирования.

Штриховой код (штрих-код) представляет собой последовательность черных (штрихов) и белых (пробелов) элементов различной ширины, расположенных по определенным правилам. Такой подход позволяет закодировать набор символов в графическом представлении и обеспечить машиночитаемый вид. Считать штрих-код и извлечь из него информацию

можно как с помощью специального оборудования (различных сканеров), так и с помощью обычных смартфонов.

Штрихкоды можно разделить на две группы: линейные и двухмерные. Линейный штрихкод – код, который читается в одном направлении, характеризуется простотой эксплуатации. Пример такого штрихкода показан на Рис. 6.

Рис. 6: Линейный штрихкод

С необходимостью кодировать все больше и больше информации широко стали использоваться двухмерные штрихкоды. Двухмерный штрихкод – код, который расшифровывается в двух измерениях: по вертикали и по горизонтали, разработан для кодирования большого объема информации. Примеры двухмерного штрихкода показаны на Рис. 7.

Рис. 7: Двухмерные штрихкоды

Одним из наиболее часто используемых двухмерных штрихкодов является QR-код (Quick Response Code). Структура QR-кода представлена на Рис. 8.

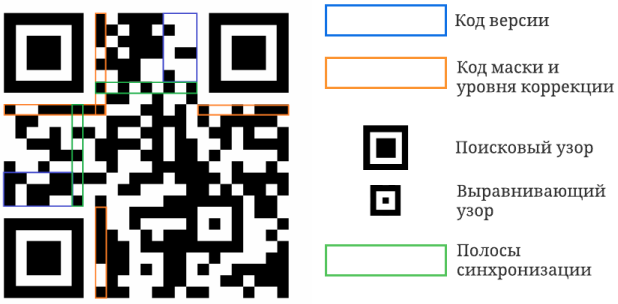

Рис. 8: Структура QR-кода

Версия QR-кода определяет его размер: коды версии 1 имеют наименьший размер – 21×21 пиксель, а коды версии 40 обладают самым большим размером – 177×177 пикселей. Код маски и уровня коррекции нужны для определения уровня коррекции ошибок. Всего их 4, их различие заключается в разных количествах полезной информации, которую можно восстановить при повреждении кода. Поисковый узор нужен для детектирования кода при его сканировании. Выравнивающий узор используется для дополнительной стабилизации кода при его декодировании. Полосы синхронизации требуются для определения размера модулей. Вокруг содержимого QR-кода расположено белое поле, которое нужно при сканировании, чтобы отличить QR-код от его окружения. Более подробно ознакомиться с технологией штрихового кодирования можно в работе [13].

1.3 Алгоритмы для встраивания QR-кодов в цветные изображения

В качестве следующего уровня при эшелонированной защите данных решено использовать метод стеганографии, то есть необходимо скрыть не только саму информацию, но и сам факт ее хранения или передачи. Для решения этой задачи будут использованы алгоритм доступа к бинарным слоям цветных изображений и алгоритм встраивания QR-кодов в цветные изображения.

При комбинации перечисленных выше алгоритмов можно получить такой алгоритм, на вход которому поступают QR-коды, представляющие какую-то зашифрованную информацию. Этот алгоритм позволяет встроить QR-коды в цветное изображение (так называемый контейнер) таким образом, что внешне изображение не меняется, но тем не менее становится носителем информации, которую потом можно из этого изображения-контейнера извлечь.

1.3.1 Алгоритм доступа к бинарным слоям цветных изображений

Как известно, цветное изображение размера N × M с тремя цветовыми компонентами R (Red), G (Green), B (Blue) можно представить в виде матрицы N×M×3, где каждый пиксель содержит информацию о цвете – об интенсивности каждой цветовой компоненты в этом пикселе. То есть можно интерпретировать, что каждому цветовому каналу соответствует матрица N×M, где каждый элемент отражает интенсивность соответствующего цвета в этом пикселе. Идея алгоритма доступа к бинарным слоям заключается в том, что каждый цветовой канал можно разложить на восемь виртуальных слоев, так называемых виртуальных битовых срезов, которые содержат только значения «0» или «1». Виртуальными эти слои можно назвать по той причине, что физически они никак не отображены в цветовых компонентах, а лишь отражают границы восьми уровней значений интенсивности цвета в пикселе компоненты изображения (Рис. 9).

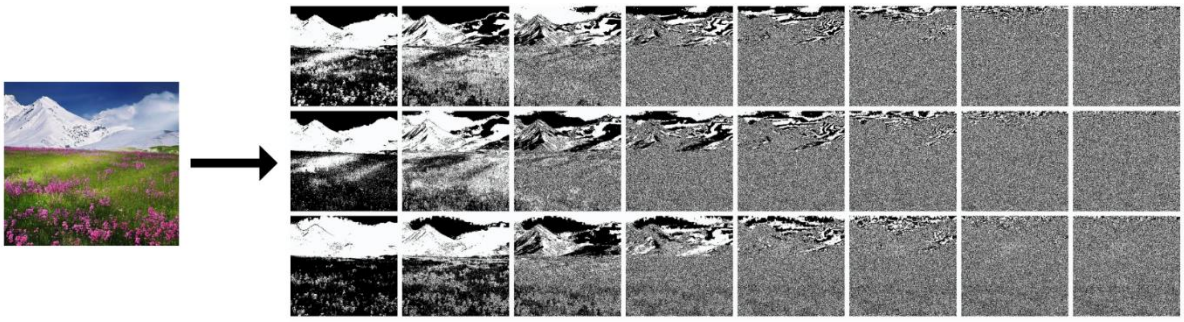

Рис. 9: Разложение изображения на 8 битовых слоев для каждого цветового канала

Каждому из восьми виртуальных битовых срезов соответствует свой множитель, являющийся степенью двойки: 128, 64, 32, 16, 8, 4, 2, 1. Чем больше множитель, тем значимее битовый слой: множитель 128 соответствует наиболее значимому битовому срезу (MSB – Most Significant Bits), а множитель 1 – наименее значимому битовому слою (LSB – Least Significant Bits). Описанным образом можно отразить любое значение интенсивности от 0 до 255. Например, если интенсивность цвета в этом пикселе равна 255, то

значения во всех битовых срезах равны «1», так как $128 \times 1 + 64 \times 1 + 32 \times 1 + 16 \times 1 + 8 \times 1 + 4 \times 1 + 2 \times 1 + 1 \times 1 = 255$.

1.3.2 Алгоритм встраивания QR-кодов в цветные изображения

Алгоритм встраивания QR-кодов основан на описанном выше алгоритме доступа к бинарным слоям. Для встраивания QR-кодов необходимо получить доступ к наименее значимым битовым слоям каждой цветовой компоненты, обнулить область слоя необходимого размера и сложить с соответствующим QR-кодом, то есть в результате слой будет содержать как QR-код, так и часть изначальной информации. Так как затрагиваются незначимые слои изображения, встраивание QR-кодов в эти срезы лишь немного меняет изображение, то есть модификации изображения внешне незаметны с первого взгляда.

В зависимости от задачи можно встраивать коды в разное количество последних битовых слоев. Можно использовать лишь последний битовый срез (LSB-слой), в таком случае в изображение-контейнер можно встроить три QR-кода (по одному в последний слой каждого цветового канала). Можно использовать два последних битовых среза, тогда можно встроить шесть QR-кодов (Рис. 10). Но стоит отметить, что изменения в большом количестве битовых срезов могут исказить изображение, то есть встроенные QR-коды могут стать заметны, если будут записаны в значимые битовые слои.

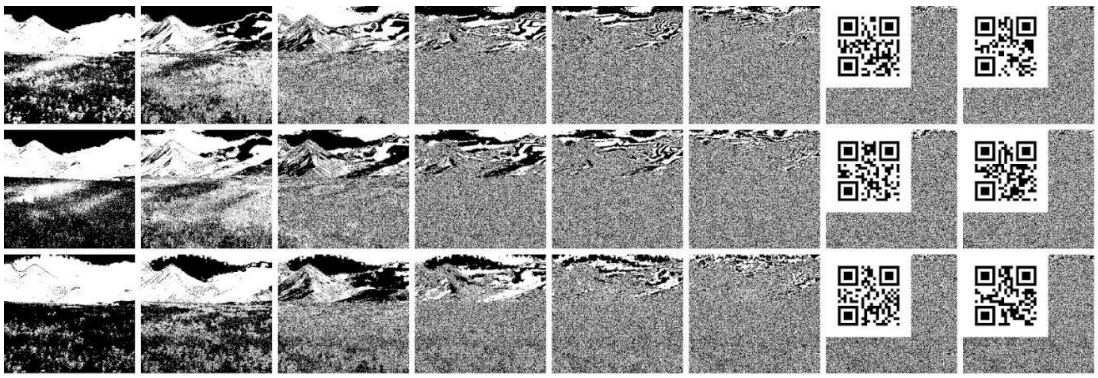

Рис. 10: Бинарные слои изображения после встраивания QR-кодов

После применения алгоритма изображение становится носителем информации. С помощью соответствующего декодирующего алгоритма

можно извлечь встроенные в изображение QR-коды для считывания информации.

Таким образом, некие данные можно преобразовать в QR-код с помощью соответствующих генераторов. Затем с помощью первого алгоритма доступа к бинарным срезам можно получить доступ к LSB-слоям компонент “Red”, “Green”, “Blue” в цветном изображении-контейнере, выбранном для хранения информации. И впоследствии с применением второго алгоритма встраивания QR-кодов можно «обнулить» LSB-слои компонент и «записать» в них соответствующие QR-коды с необходимыми данными. В результате получается изображение, которое является носителем каких-то встроенных в него данных.

Чтобы убедиться в том, что изображение после таких преобразований не сильно меняется, можно рассмотреть разность оригинального изображения и изображения со встроенными QR-кодами (Рис. 11).

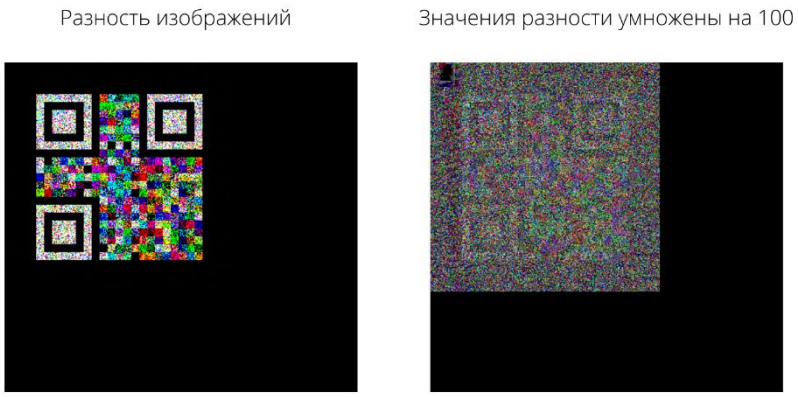

Рис. 11: Разность оригинала изображения и результата применения алгоритма

Видно, что изображения различаются в верхнем левом углу, то есть в той области, куда встраиваются QR-коды. Черная область означает, что в этом месте изображения совпадают.

На Рис. 12 представлены цветные битовые слои оригинального изображения, изображения со встроенными QR-кодами и их разности. Можно заметить, что изображения совпадают в большинстве слоев и различаются лишь в наименее значимых слоях в тех местах, куда записывается QR-код.

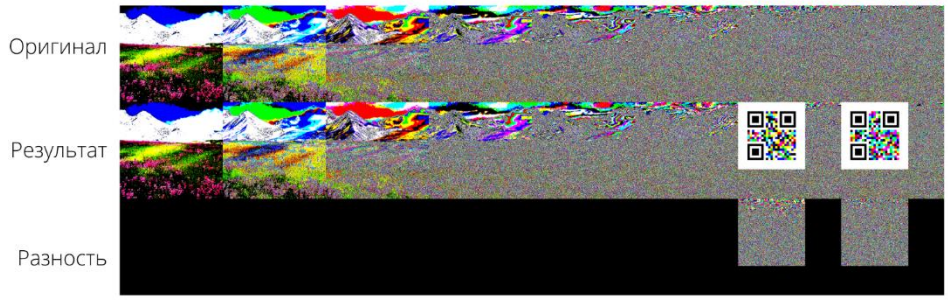

Рис. 12: Цветные битовые слои оригинального изображения, изображения со встроенными QR-кодами и их разности

На Рис. 13 можно увидеть битовые слои трех цветовых компонент R, G, B разности изображений.

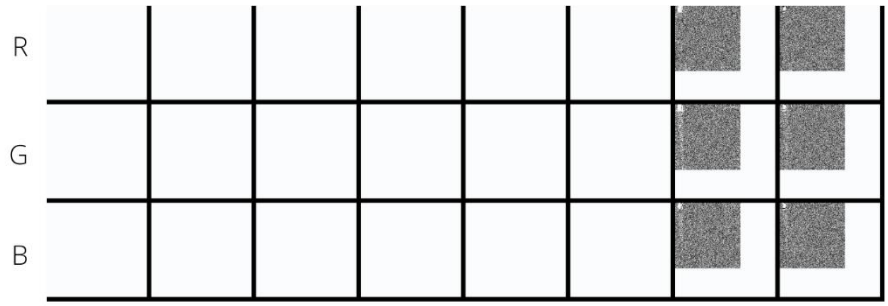

Рис. 13: Битовые слои трех цветовых каналов разности изображений

На Рис. 14 представлены исходное изображение и результат работы алгоритма встраивания QR-кодов при затрагивании двух последних битовых слоев.

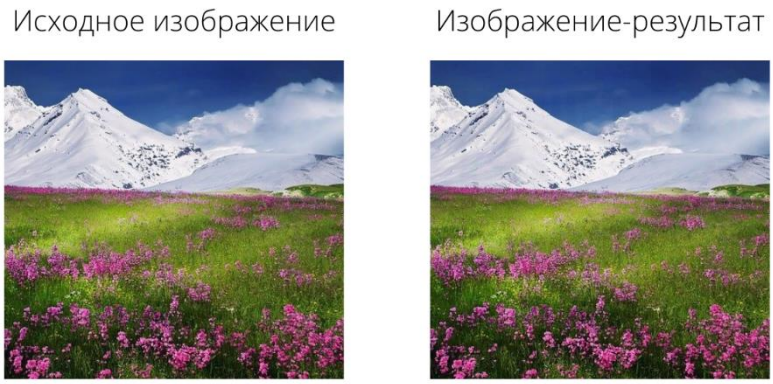

Рис. 14: Исходное изображение и изображение-результат

Глава 2. Описание предлагаемого метода кодирования данных

В предыдущей главе были описаны различные алгоритмы для кодирования данных: алгоритм шифрования AES, преобразование информации в QR-код, алгоритм доступа к бинарным слоям цветных изображений и алгоритм встраивания QR-кодов в цветные изображения. В этой главе будут описаны методы кодирования и декодирования данных, составленные на основе этих алгоритмов.

2.1 Метод кодирования данных

Метод кодирования данных состоит из описанных выше алгоритмов. Кодирование данных будет происходить следующим образом:

1. Криптографическое шифрование медицинских данных о результатах обследования алгоритмом AES.
2. Преобразование зашифрованной информации в QR-код.
3. Получение доступа к LSB-слоям (LSB – Least Significant Bits) компонент “Red”, “Green”, “Blue” в цветном изображении-контейнере, выбранном для хранения информации, с помощью алгоритма доступа к бинарным слоям цветных изображений.
4. «Обнуление» LSB-слоев компонент, то есть удаление содержимого изображения в этих слоях, и «запись» в них QR-кодов с необходимыми данными.

В результате будет получено изображение, которое будет носителем результатов медицинских обследований. Схема встраивания данных в цветное изображение показана на Рис. 15. Такой метод кодирования позволяет сохранять и обмениваться данными и регулировать доступ к хранимой информации. Достигается «многослойная» защита данных.

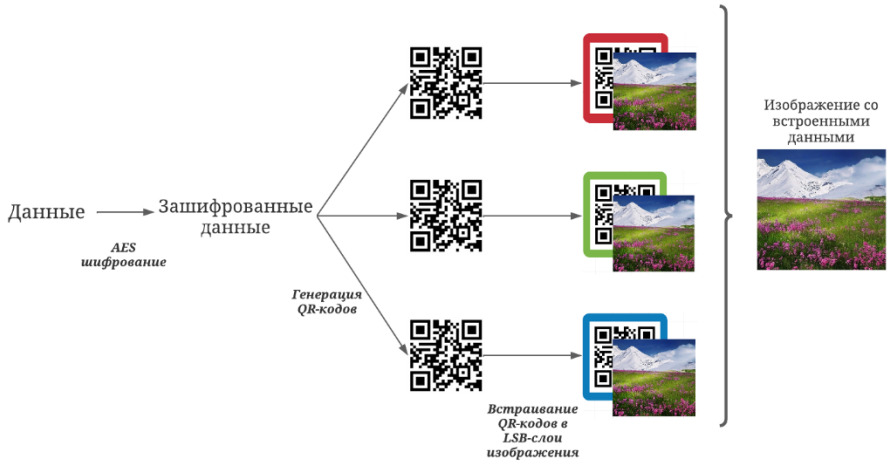

Рис. 15: Метод кодирования данных

2.2 Метод декодирования данных

Для того, чтобы прочитать данные из изображения-контейнера, нужно применить описанные выше преобразования в обратном порядке:

1. Получение доступа к LSB-слоям и хранящимся в них QR-кодам с помощью алгоритма доступа к бинарным слоям цветных изображений.
2. Извлечение QR-кодов из LSB-слоев изображения.
3. Преобразование информации из графического представления в виде QR-кода в символьный вид.
4. Применение алгоритма дешифрования AES.

Схема обратного метода представлена на Рис. 16.

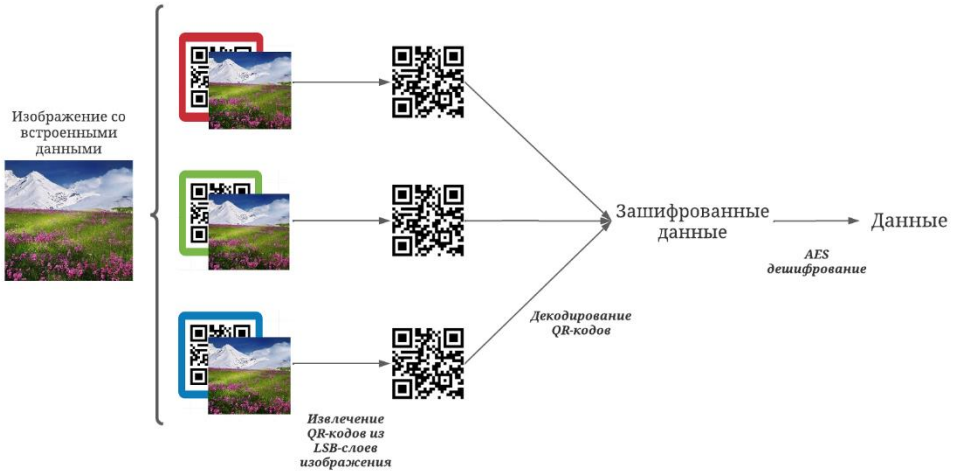

Рис. 16: Метод декодирования данных

Глава 3. Разработка приложения

В предыдущей главе был описан метод кодирования данных. Для его исполнения и для достижения цели данной работы, а именно для реализации эффективного метода для безопасного хранения и передачи данных, необходимо создать приложение с графическим пользовательским интерфейсом. В данной главе будет проведено проектирование и реализация соответствующего приложения.

3.1 Общее описание приложения для кодирования данных

Главной задачей приложения является реализация методов кодирования и декодирования данных, описанных ранее.

Процесс запуска метода в приложении состоит из следующих этапов:

1. Преобразование формата данных.
2. Ввод необходимых параметров для работы алгоритмов криптографического шифрования и встраивания QR-кодов в изображение или криптографического дешифрования и извлечения данных из изображения.
3. Вызов алгоритмов для кодирования или декодирования.
4. Сохранение результатов.

Исходные данные, содержащие информацию о пациентах, либо сразу могут быть использованы для кодирования, либо нуждаются в предобработке и в преобразовании в формат, подходящий для последующих этапов работы метода. В обратном методе извлеченные данные должны быть преобразованы в исходный формат.

Для работы метода кодирования необходимо ввести ключ шифрования для алгоритма AES и выбрать изображение-контейнер, который будет использован для встраивания в него данных, для метода декодирования нужен только ключ шифрования.

С использованием введенных параметров происходит кодирование или декодирование данных по описанному методу. Результат сохраняется в выбранную директорию.

У приложения должен быть графический интерфейс пользователя для метода кодирования и интерфейс для обратного метода.

Для программной реализации приложения решено использовать язык программирования Python [14], так как он позволяет повысить скорость разработки за счет своей простоты и обладает разнообразием библиотек: для работы с изображениями (PIL (Python Image Library)), QR-кодами (qrcode), для работы с массивами (NumPy), для работы с табличными данными (Pandas) и для создания графического интерфейса (Tkinter). Для разработки использована интегрированная среда разработки PyCharm.

3.2 Сценарии использования

Для проектирования приложения необходимо сформулировать пользовательские требования в виде сценариев использования. Пользовательские требования описывают цели или задачи, которые пользователи должны иметь возможность выполнять с помощью продукта, который в свою очередь должен приносить пользу [15].

В приложении выделена одна роль – пользователь. Пользователями приложения будут сотрудники НМИЦ ПН им. В. М. Бехтерева. Диаграмма сценариев использования представлена на Рис. 17.

Рис. 17: Диаграмма сценариев использования

Сценарий «Запустить метод кодирования» включает в себя следующие действия:

1. Выбрать файл с данными, которые будут закодированы.
2. Определить входные параметры для метода кодирования: изображение-контейнер, в которое будут внедрены данные, и ключ шифрования.
3. Выбрать параметры для сохранения результата: директорию и название для результата.

Сценарий «Запустить метод декодирования» содержит следующие возможности:

1. Выбрать изображение, в которое были встроены данные.
2. Ввести ключ шифрования.
3. Выбрать параметры для сохранения результата: директорию и название для результата.

3.3 Преобразование формата данных

В приложении входные данные могут быть двух типов: текстового формата .txt и Excel-файлы формата .xlsx, .xls. Файлы текстового формата не нуждаются в предварительной обработке, данные из них могут быть извлечены и переданы для кодирования.

Использование Excel-файлов является более распространенным вариантом, так как этот формат используется в НМИЦ ПН им. В. М. Бехтерева для хранения карты пациента. Такие файлы должны быть подвергнуты предобработке и преобразованию формата.

Карта пациента в Excel-файле содержит несколько блоков: социально-демографические характеристики (ФИО, дата рождения и т.д.), анамнез, анализы (кровь), экспериментально-психологическое обследование. Каждый столбец в табличном файле соответствует определенному показателю. Всего задействовано 144 столбца. Запись из этого файла необходимо преобразовать в обычный текст, в строку, чтобы можно было применить к ней алгоритм шифрования AES.

Преобразование происходит путем составления из табличной записи строки, похожей по структуре на словарь, где ключами выступают названия столбцов, а значениями – значения из этих столбцов, разные пары ключ-значение разделены знаком «;». Получается строка вида «<название столбца 1>:<значение столбца 1>;<название столбца 2>:<значение столбца 2>;<название столбца 3>:<значение столбца 3>;».

3.4 Архитектура приложения

Приложение для кодирования и декодирования данных состоит из нескольких модулей, каждый из которых предназначен для решения конкретной задачи.

Программный комплекс содержит 8 модулей:

- модуль преобразования формата данных,
- модуль шифрования AES,
- модуль дешифрования AES,
- модуль подбора подходящего изображения-контейнера,
- модуль встраивания QR-кодов в цветное изображение,
- модуль извлечения данных из цветного изображения,
- модуль графического интерфейса пользователя для метода кодирования,
- модуль графического интерфейса пользователя для метода декодирования.

Модуль преобразования формата данных предназначен для извлечения необходимых данных из Excel-файла и формирования строки определенного шаблона, передаваемой на вход модулю шифрования, а в случае метода декодирования данных для преобразования строки определенного шаблона в Excel-файл. Модуль представлен в файле «transform_file_format.py» и содержит 2 функции:

1. Функция «from_excel_to_format» предназначена для преобразования данных из Excel-файла в строку вида «<название столбца

1>:<значение столбца 1>;<название столбца 2>:<значение столбца 2>;<название столбца 3>:<значение столбца 3>;».

2. Функция «from_format_to_excel» выполняет обратное преобразование: из строки приведенного выше вида формируется Excel-файл.

Для работы с Excel-файлами использована библиотека Pandas.

Модуль шифрования AES предназначен для применения криптографического шифрования AES к данным, что в описанном в главе 2 методе кодирования данных соответствует шагу 1. Содержится в файле «aes_cipher.py». Главная функция «aes_algorithm» делит входную строку на блоки размером 16 байт и для каждого блока вызывает функцию криптографического шифрования «cipher», которая согласно алгоритму AES выполняет 10 раундов различных преобразований, то есть вызывает следующие функции: «addRoundKey», «subBytes», «shiftRows», «mixColumns».

Модуль дешифрования AES предназначен для дешифрования данных, к которым ранее был применен алгоритм AES. Соответствует шагу 4 метода декодирования данных в главе 2. Хранится в файле «aes_decipher.py». Аналогично предыдущему модулю главная функция «algorithmInv» делит входные данные на блоки по 16 байт и применяет к ним функцию дешифрования «decipher», которая в рамках раундов вызывает функции обратных преобразований: «addRoundKey», «shiftRowsInv», «subBytesInv», «mixColumnsInv».

Модуль подбора подходящего изображения-контейнера содержится в файле «pick_img.py». Так как от размера зашифрованных данных зависит размер формируемых QR-кодов, которые будут встроены в слои изображения, необходимо на основе размера зашифрованных данных рассчитать размер изображения, в которое эти данные могут поместиться.

Задано несколько значений возможных размеров. Вычисленный размер округляется в большую сторону до ближайшего значения. Для каждого

заданного размера имеется набор изображений, из которых пользователь может выбрать одно.

Модуль встраивания QR-кодов в цветное изображение предназначен для формирования QR-кодов по зашифрованной информации и их встраиванию в цветное изображение-контейнер. Соответствует исполнению шагов 2, 3, 4 метода кодирования данных из главы 2. Представлен в файле «binary_layers.py». В главной функции модуля «algorithm» происходит формирование QR-кодов из зашифрованных данных и вызов функций, реализующих алгоритм доступа к бинарным слоям цветного изображения («get_binary_layers») и алгоритм встраивания QR-кодов в цветное изображение («embed_qr_into_image»).

В модуле использованы библиотеки PIL, qrcode и NumPy.

Модуль извлечения данных из цветного изображения предназначен для извлечения QR-кодов из цветного изображения и их преобразования в символьный вид. Соответствует реализации шагов 1, 2, 3 метода декодирования данных. Хранится в файле «decode.py». Функция «algorithm» вызывает функцию извлечения QR-кодов из LSB-слоев изображения («get_qr_from_lsb») и декодирует данные из QR-кодов в символьный вид. В модуле так же использованы библиотеки PIL, qrcode и NumPy.

Модуль графического интерфейса пользователя для метода кодирования и модуль графического интерфейса пользователя для метода декодирования предназначены для реализации графических интерфейсов пользователя и для демонстрации результата работы методов.

Архитектура программного комплекса, состоящего из описанных выше модулей, представлена на Рис. 18.

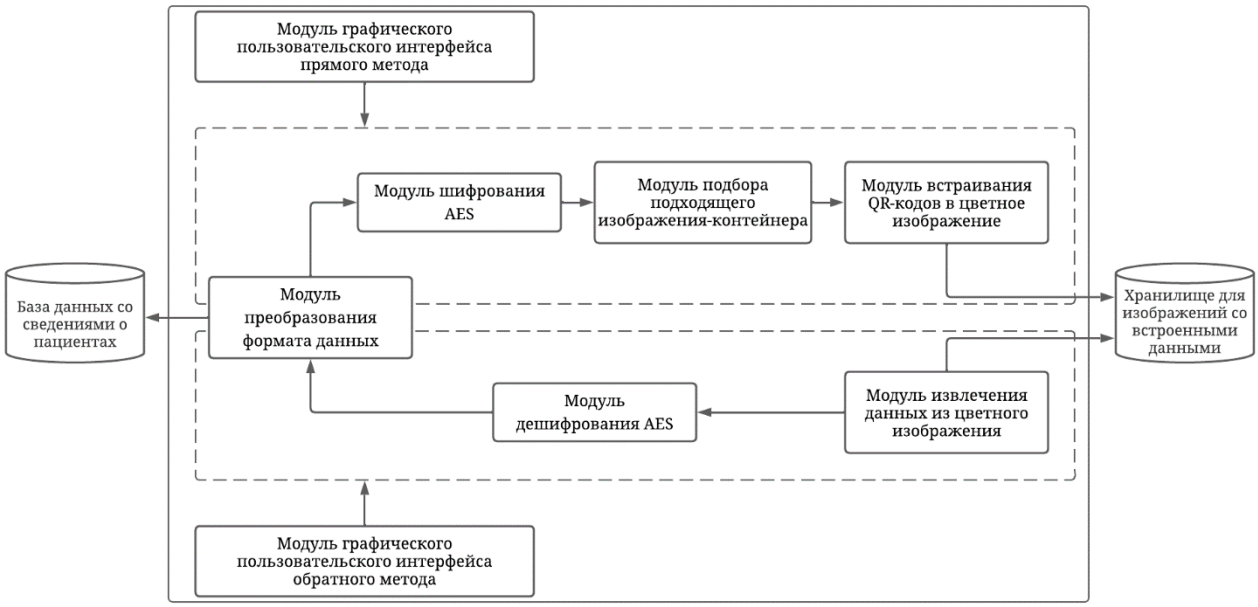

Рис. 18: Диаграмма модулей приложения

3.5 Графический пользовательский интерфейс

Приложение содержит два графических интерфейса пользователя: для метода кодирования данных и для метода декодирования данных.

Главное требование к оформлению графических интерфейсов: они должны быть интуитивно понятны. Чтобы этого добиться, определены следующие требования к стилю интерфейсов:

- понятность и доступность,
- минимализм, то есть отсутствие нагромождения элементов,
- сочетание контрастных цветов для лучшей видимости элементов.

Графические интерфейсы пользователя созданы с использованием библиотеки Tkinter. Каждый из интерфейсов содержит одно главное окно, содержащее поля для ввода параметров, и всплывающее окно для показа результата работы метода.

Главное окно интерфейса для метода кодирования представлено на Рис. 19. Оно содержит:

- поле для выбора необходимого файла, содержащего данные о пациенте,
- поле для определения формата входного файла,

- кнопку для запуска модуля подбора подходящих изображений, которые могут быть использованы как контейнер,
- поле для ввода ключа шифрования AES,
- поле выбора директории для сохранения результата работы метода,
- поле ввода названия для результирующего изображения,
- кнопку для запуска метода кодирования.

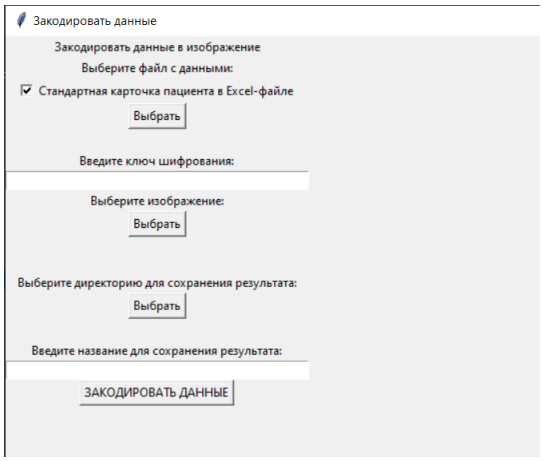

Рис. 19: Главное окно графического интерфейса метода кодирования

После заполнения всех полей и запуска метода кодирования путем нажатия на кнопку «Закодировать данные» появляется отдельное окно с выводом результата в случае корректной работы метода (Рис. 20) или сообщение об ошибке в случае некорректной работы метода.

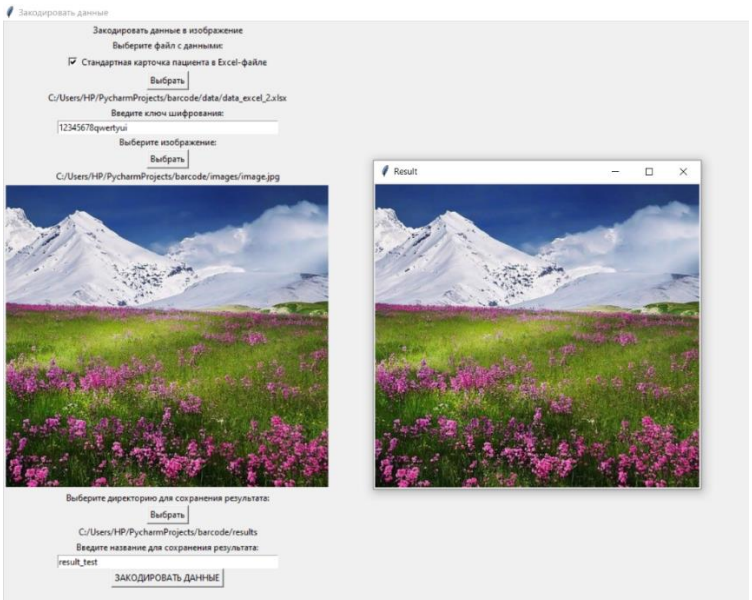

Рис. 20: Вывод результата в графическом интерфейсе метода кодирования

Главное окно интерфейса для метода декодирования после заполнения всех полей показано на Рис. 21. Требования к содержанию главного окна:

- поле выбора изображения, которое является носителем информации,
- поле для определения формата результирующего файла,
- поле ввода ключа шифрования,
- поле для выбора директории для сохранения результата,
- поле ввода названия для файла-результата,
- кнопка запуска метода декодирования.

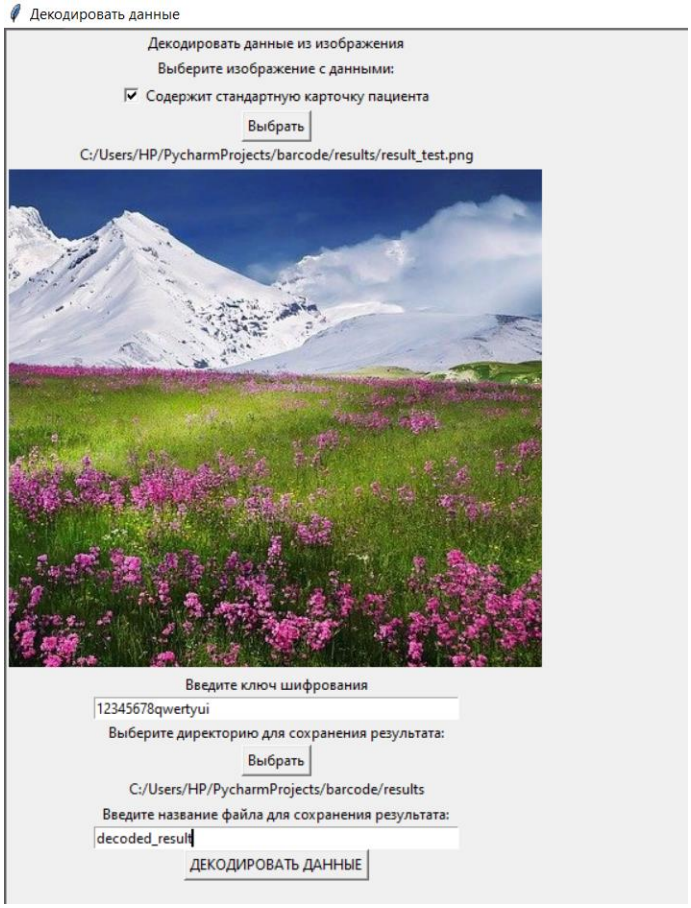

Рис. 21: Главное окно графического интерфейса метода декодирования после заполнения полей

После нажатия на кнопку «Декодировать данные» результат открывается в отдельном окне. Так как была извлечена стандартная карта пациента, результат сохраняется в виде Excel-файла, поэтому для вывода результата открывается окно Excel (Рис. 22), а в случае возникновения ошибки должно выводиться соответствующее уведомление.

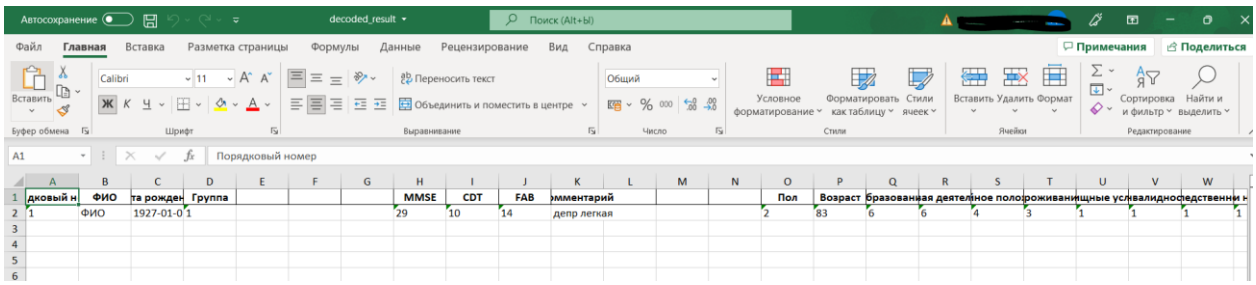

Рис. 22: Вывод результата для метода декодирования

Глава 4. Эксперименты и анализ метода кодирования данных

Для улучшения работы метода кодирования данных важно провести эксперименты для его тестирования и анализа.

4.1 Проверка корректной работы метода в приложении

Для тестирования корректной работы приложения и методов было проведено сравнение исходных данных метода кодирования и результата метода декодирования. Карта пациента, хранимая в виде Excel-файла, была закодирована в изображение по методу, а затем были применены обратные алгоритмы. Информация, данная на вход прямому методу, совпадает с расшифрованной обратным методом информацией, то есть можно сделать вывод о корректной работе методов и приложения, которое их реализует.

4.2 Преобразования для улучшения работы метода

Эксперименты, проведенные для анализа метода кодирования, необходимы для улучшения его работы. Нужно добиться того, чтобы выполнялся главный принцип стеганографии – сокрытие информации таким образом, чтобы скрывался и сам факт наличия информации в носителе. В данном методе это означает, что изображение со встроенными данными должно выглядеть как исходное, чтобы не было подозрений о наличии встроенных данных.

Преобразования метода для улучшения его работы направлены на то, чтобы в процессе кодирования изображение затрагивалось и менялось как можно меньше. В исходном алгоритме встраивания QR-кодов в цветное изображение LSB-слои обнуляются полностью перед записью в них QR-кодов, то есть пропадает часть информации об изображении (Рис. 23).

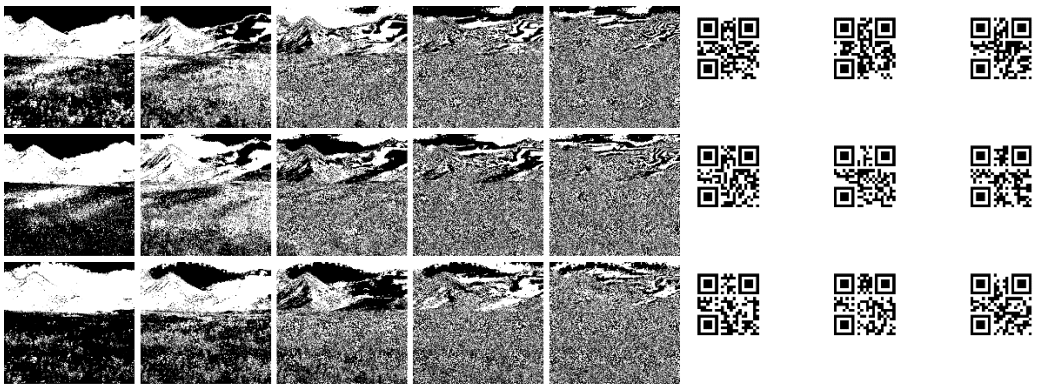

Рис. 23: Бинарные слои при полном обнулении перед встраиванием QR-кодов

Для устранения этой проблемы решено обнулять и использовать только ту часть LSB-слоя, в которую будет записан QR-код (Рис. 24).

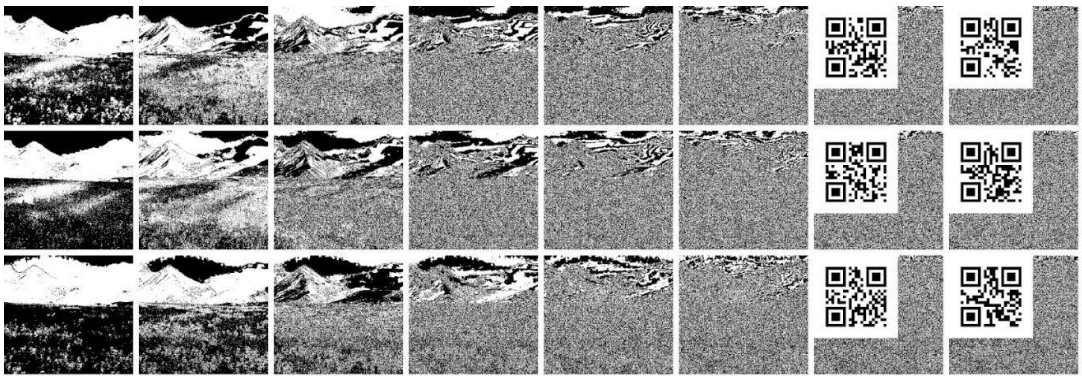

Рис. 24: Бинарные слои при обнулении только нужной части для встраивания QR-кода

Чтобы вносить минимальные изменения в изображение, но при этом помещать в него необходимый объем информации, нужно выбрать оптимальное количество слоев, которые будут затронуты. На Рис. 25 показано сравнение результатов при разном количестве задействованных слоев.

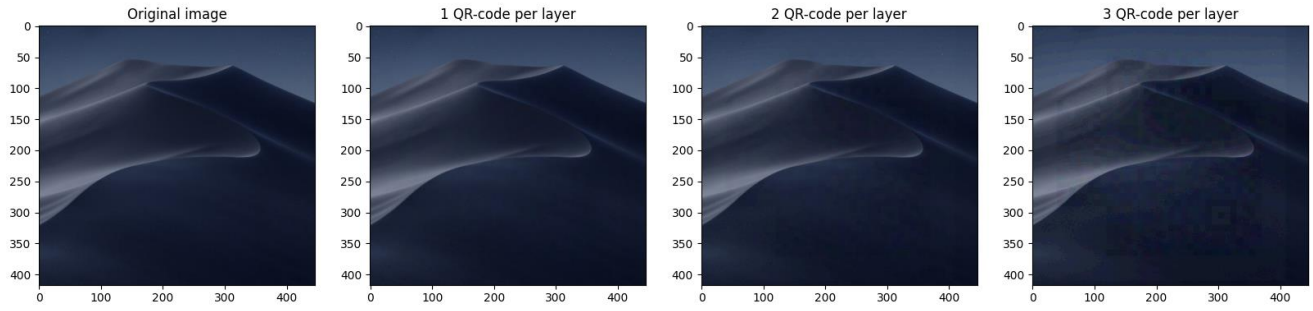

Рис. 25: Сравнение результатов при разном количестве затрагиваемых слоев

При использовании 9 слоев (по 3 слоя в каждой цветовой компоненте) изображение искажается и заметны встроенные QR-коды, потому что затрагиваются значимые слои. При использовании 6 (по 2 слоя в каждой

цветовой компоненте) и 3 (по 1 слою в каждой цветовой компоненте) слоев изменения изображения гораздо менее заметны, но при этом в 6 слоев можно вместить больше данных, чем в 3 слоя, поэтому в методе решено задействовать 6 слоев как оптимальный вариант для вместимости и незаметного преобразования изображения.

В ходе проведения экспериментов было замечено, что на некоторых изображениях после встраивания данных можно заметить черные поисковые узоры QR-кода, так как они накладываются друг на друга. Для хранения самих данных эти элементы не используются, поэтому решено удалять их из QR-кодов перед их встраиванием в изображение в прямом методе (Рис. 26) и добавлять обратно после извлечения из изображения в обратном методе.

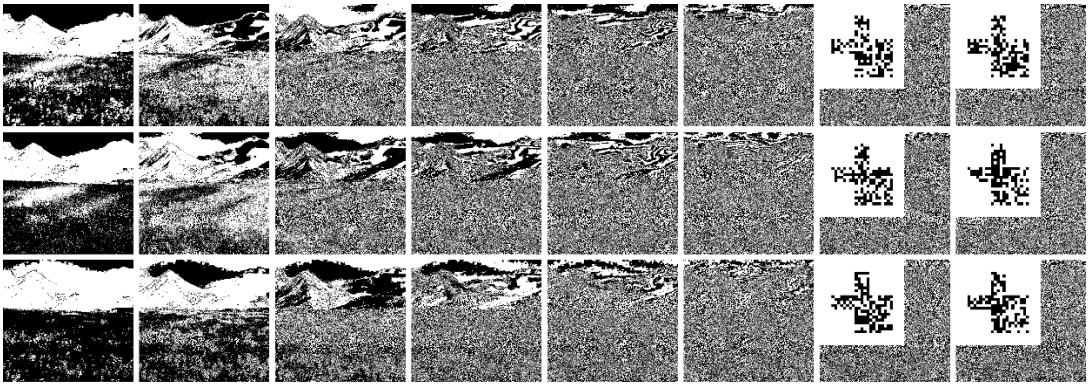

Рис. 26: Бинарные слои при удалении поисковых узоров из QR-кодов

4.3 Выбор подходящего изображения-контейнера

Выбор самого изображения так же влияет на то, насколько заметны будут встроенные QR-коды. Если изображение однотонное и тусклое, то встроенные QR-коды будут заметны. Если изображение яркое, контрастное и содержит много разных цветов, то затрагивание слоев будет незаметно.

Сравнение результатов при использовании разных изображений приведено на Рис. 27. Первое изображение однотонное, поэтому получен плохой результат, при котором встроенные QR-коды очень заметны. Второе изображение более яркое и контрастное, результат получше, но все равно можно заметить искажения. Самый хороший результат получен для третьего

изображения, на котором нет однотонных объектов и много разных деталей, поэтому изменения не видны.

Рис. 27: Сравнение результатов при использовании разных изображений

Различие этих изображений можно увидеть на гистограмме (Рис. 28). У изображения с плохим результатом на гистограмме можно увидеть, что цвета сконцентрированы в начале оси абсцисс, то есть изображение темное и тусклое. У изображения со средним результатом цвета распределены по всему диапазону, но при этом цвета распределены неравномерно. У изображения с хорошим результатом цвета распределены почти одинаково, график более гладкий, чем в других изображениях.

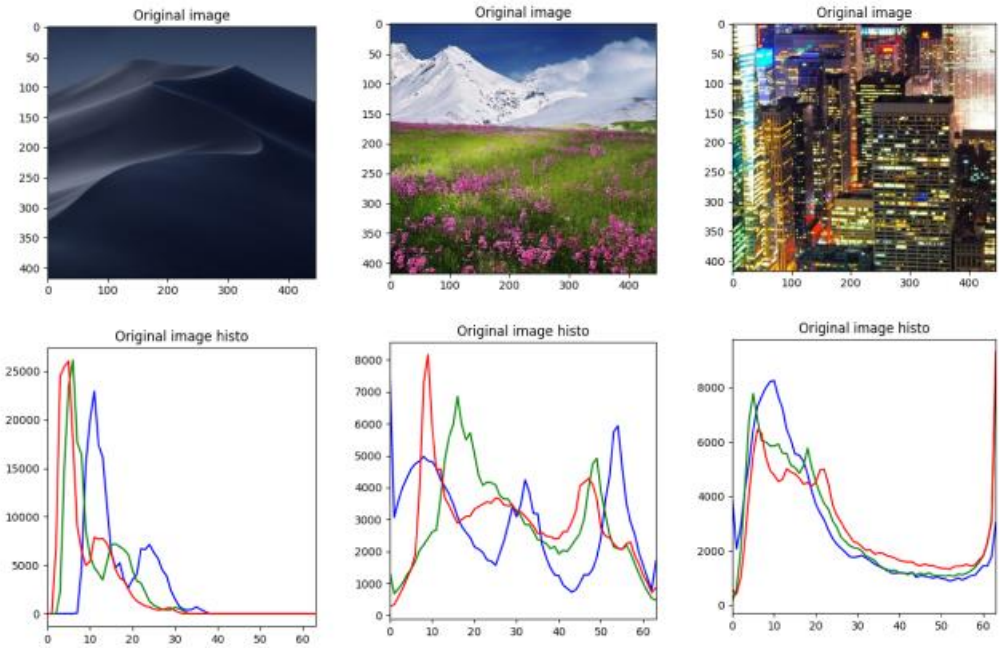

Рис. 28: Сравнение гистограмм изображений

Выводы

В этой работе был составлен метод многослойной защиты данных, который позволяет безопасно хранить и передавать персональные данные. Поставленные задачи были выполнены. Были изучены различные подходы и был составлен метод, который обеспечивает кодирование данных таким образом, что неизвестно, что в носителе информации содержатся какие-то сведения. Технология представляет из себя кодирование данных, состоящее из нескольких этапов. Сначала данные подвергаются криптографическому шифрованию AES, затем из них формируются QR-коды, которые с помощью алгоритма доступа к бинарным слоям и алгоритма встраивания QR-кодов в цветное изображение помещаются в изображение-контейнер. Спроектировано и реализовано приложение для реализации метода кодирования и декодирования информации с графическим интерфейсом пользователя. Были проведены эксперименты, которые позволили убедиться в корректной работе метода и приложения, а также проанализировать метод и внести в него изменения для более незаметного внедрения данных в изображение.

Заключение

Реализованная технология для кодирования обеспечивает защиту персональных данных, что особенно актуально для медицинских учреждений. Технология реализует многослойную защиту данных, что обеспечивает ее надежное сокрытие. Сначала данные подвергаются криптографическому шифрованию, а затем в виде QR-кодов встраиваются в LSB-слои цветного изображения. При таком подходе достигается принцип стеганографии, то есть полученное изображение со встроенными данными не отличается от исходного, поэтому без специальных знаний нельзя узнать, что оно является носителем информации.

Отмеченные характеристики позволяют использовать этот метод для работы с персональными данными, которые необходимо надежно защищать.

В закодированном виде данные можно безопасно хранить и передавать по различным каналам связи. Это поможет снизить риски утечки медицинских данных, что в настоящее время является актуальной проблемой, ведь даже если закодированные данные попадут в руки третьим лицам, злоумышленники не смогут ее извлечь без наличия ключа шифрования.

Разработанное приложение обладает простым и понятным интерфейсом и готово к использованию в НМИЦ ПН им. В. М. Бехтерева. С его помощью можно надежно закодировать данные о пациенте в цветное изображение или извлечь из изображения-контейнера встроенные в него сведения.

Список использованных источников

1. Риски и последствия утечки информации. [Электронный ресурс]: URL:https://dlbi.ru/data-base-leaks-risks/ (Дата обращения: 29.04.2022)
2. Факторы стигматизации лиц с психическими расстройствами: методические рекомендации / Ястребов В.С., Михайлова И.И., Гонжал О.А., Трущелёв С.А.; Науч. центр психического здоровья РАМН. – М., Изд-во ЗАО Юстицинформ 2009. – 22 с.
3. Причины утечки информации. [Электронный ресурс]: URL:https://searchinform.ru/analitika-v-oblasti-ib/utechki-informatsii/prichiny-utechki-informatsii/ (Дата обращения: 29.04.2022)
4. Федеральный закон от 27.07.2006 N 152-ФЗ (ред. от 02.07.2021) "О персональных данных". [Электронный ресурс]: URL:http://www.consultant.ru/document/cons_doc_LAW_61801/4f41fe599ce341751e4e34dc50a4b676674c1416/ (Дата обращения: 29.04.2022)
5. Волчинская, Е.К. Персональные данные в России 2010 [Текст] / Е.К. Волчинская // Защита персональных данных. Опыт правового регулирования. - 2010. - № 6. - С. 5-7.
6. Белышев Д. В., Кочуров Е. В. Анализ методов хранения данных в современных медицинских информационных системах //Программные системы: теория и приложения. – 2016. – Т. 7. – №. 2 (29). – С. 85-103.
7. Stytz M. R. Considering defense in depth for software applications //IEEE Security & Privacy. – 2004. – Т. 2. – №. 1. – С. 72-75.
8. Орлов П. О. Исследование функциональных возможностей асимметричных алгоритмов шифрования //Евразийский научный журнал. – 2015. – №. 8.
9. Rihan S. D., Khalid A., Osman S. E. F. A performance comparison of encryption algorithms AES and DES //International Journal of Engineering Research & Technology (IJERT). – 2015. – Т. 4. – №. 12. – С. 151-154.

10. Penchalaiah N., Seshadri R. Effective Comparison and evaluation of DES and Rijndael Algorithm (AES) //International journal of computer science and engineering. – 2010. – Т. 2. – №. 05. – С. 1641-1645.
11. Казиева, Н. Методы и алгоритмы штрихового кодирования для задач лицевой биометрии : диссертация на соискание ученой степени кандидата технических наук. – Национальный исследовательский университет ИТМО. – Санкт-Петербург, 2020.
12. Buchanan, William J. Cryptography. – Edinburgh: River Publishers, 2017. – 350 p.
13. Востриков А. А., Сергеев М. Б. Штриховое кодирование. Учебное пособие //СПб.: ГУАП. – 2011.
14. Язык программирования Python 3.10. [Электронный ресурс]: URL:https://www.python.org/doc/ (Дата обращения: 29.05.2022)
15. Вигерс К., Битти Д. Разработка требований к программному обеспечению //М.: Русская редакция. – 2004.