МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ ИМЕНИ ГУМАРБЕКА ДАУКЕЕВА»
Институт Систем Управления и Информационных Технологии
Кафедра «Системы информационной безопасности»

«ДОПУЩЕН К ЗАЩИТЕ»
Зав.кафедрой к.п.н., доцент Бердибаев Рат Шындалиевич
(ученая степень, звание, Ф.И.О.)

_____ «8» ___июня_____2020г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: «NFC-технологии в механизмах аутентификации»_____

Специальность Системы Информационной Безопасности_____

Выполнил(а) Ефименко Никита Сергеевич_____ Группа СИБу-17-3___
(Ф.И.О.)

Научный руководитель к.т.н. доцент Сатимова Елена Григорьевна,_____
(ученая степень, звание, Ф.И.О.)

ст. преп. каф. СИБ Зуева Екатерина Александровна _____

Консультанты:
по специальной части:
ст. преп. каф. СИБ Дмитриева Маргарита Валерьевна_____
_____ «_31___» __мая_____2020г.
(подпись)

по безопасности жизнедеятельности:
д.х.н., профессор Приходько Николай Георгиевич_____
_____ «_27___» __мая_____2020г.
(подпись)

Нормоконтролер: ст. преп. каф. СИБ Дмитриева Маргарита Валерьевна_____
(ученая степень, звание, Ф.И.О.)

_____ «_____» _____20____г.
(подпись)

Рецензент: старший научный сотрудник РГП «Институт информационных и
(ученая степень, звание, Ф.И.О.)
вычислительных технологий» Комитета наук МОН РК, PhD Шаяхметова Асем Серикбаевна_____

_____ «_____» _____20____г.
(подпись)

Алматы 2020

Задание на выполнение дипломного проекта

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество

«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ ИМЕНИ ГУМАРБЕКА ДАУКЕЕВА»

Институт Систем Управления и Информационных Технологии

Кафедра «Системы Информационной Безопасности»

Специальность «Системы Информационной Безопасности»

ЗАДАНИЕ

на выполнение дипломного проекта

Студенту Ефименко Никите Сергеевичу
(Ф.И.О.)

Тема проекта «NFC-технологии в механизмах аутентификации»

Утверждена приказом по университету № 563 от «30» апреля 2020 г.

Срок сдачи законченного проекта «5» июня 2020 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): среды разработки «Arduino IDE», «Microsoft Visual Studio 2017», «Android Studio», контроллер Arduino NANO и модуль RFID/NFC PN532, реестр ОС для внедрения тестовых изменений, тестовый образец для проведения отладки и демонстрации проекта.

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта: изучение теоретического материала по тематике NFC; построение компактного чертежа схемы устройства и написание корректного, отказоустойчивого кода для прошивки устройства на Arduino IDE; разработка сопровождающих устройство программ в средах Visual Studio (C#) и Android Studio (Java); проведение финального тестирования проекта; рассчет рисков ИБ; проведение расчетов, согласно стандартам БЖД; подведение итогов.

Перечень графического материала (с точным указанием обязательных чертежей): Глава 1 содержит 3 рисунка, в главе 2 представлено 7 рисунков, в главе 3 представлено 39 рисунков, в 4 главе представлено 6 рисунков, в 5 главе представлен 1 рисунок.

Основная рекомендуемая литература: Tom Igoe, Don Coleman and Brian Jepson «Beginning NFC»; Anne-Marie Lesas and Serge Miranda «The Art and Science of NFC Programming».

Консультации по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Анали рисков информационной безопасности	старший преподаватель Дмитриева Маргарита Валерьевна	17.02.2020 – 09.05.2020	
Безопасность жизнедеятельности	к.т.н. доцент Приходько Николай Георгиевич	17.02.2020 – 09.05.2020	

График
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Изучение теоретического материала	17.02.2020 – 28.02.2020	
Рассмотрение возможных вариантов сред разработки	29.02.2020 – 10.03.2020	
Разработка устройства	11.03.2020 – 15.04.2020	
Разработка сопровождающего ПО	16.04.2020 - 18.04.2020	
Тестирование программно-аппаратного комплекса	19.04.2020 – 25.04.2020	
Расчет рисков ИБ	26.04.2020 - 28.04.2020	
БЖД	29.04.2020 - 09.05.2020	

Дата выдачи задания «30» апреля 2020г.

Заведующий кафедрой _____ (Бердибаев Рат Шындалиевич)
(подпись) (ФИО)

Научный руководитель проекта _____ (Сатимова Елена Григориевна)
(подпись) (ФИО)

Научный руководитель проекта _____ (Зуева Екатерина Александровна)
(подпись) (ФИО)

Задание принял к исполнению студент _____ (Ефименко Никита Сергеевич)
(подпись) (ФИО)

Аннотация

Дипломный проект посвящен исследованию технологии беспроводной передачи данных NFC. Показан практический способ реализации альтернативного механизма аутентификации в операционной системе Windows на основе беспроводной технологии NFC с использованием смартфона.

В части, относящейся к расчету рисков, отображены показатели безопасности некоторых активов организации до и после внедрения разработанного проекта.

Раздел безопасности жизнедеятельности описывает основные требования, предъявляемые к помещениям, в которых будет использован проект, а также показаны расчеты требуемой величины освещенности и кондиционирования помещения.

Аңдатпа

Дипломдық жоба NFC сымсыз деректер технологиясын зерттеуге арналған. Смартфонды қолдана отырып NFC сымсыз технологиясына негізделген Windows операциялық жүйесінде балама аутентификация механизмін енгізудің практикалық әдісі көрсетілген.

Тәуекелдерді есептеуге қатысты бөлігінде стандартты ұйымның кейбір активтерінің қауіпсіздік көрсеткіштері әзірленген жобаны іске асырғанға дейін және одан кейін көрсетіледі.

Тіршілік әрекетінің қауіпсіздігі бөлімі жоба қолданылатын бөлмелерге қойылатын негізгі талаптарды сипаттайды, сонымен қатар жарықтандыру мен ауаны салқындатудың қажетті мөлшерін есептеу көрсетілген.

Annotation

The graduation project is dedicated to the study of NFC wireless data technology. A practical method of implementing an alternative authentication mechanism in a Windows operating system based on NFC wireless technology using a smartphone is shown.

In the part related to risk calculation, safety indicators of some assets of a standard organization are displayed before and after the implementation of the developed project.

The life safety section describes the basic requirements for the rooms in which the project will be used, as well as the calculations of the required amount of lighting and air conditioning are shown.

Содержание

Введение

Беспроводные технологии начинают охватывать все большее количество сфер. Применение им находится в банковских транзакциях, простом обмене данными между устройствами, реализации зарядных станций для портативных устройств. Большинство из этих технологий берут свое начало от технологии как RFID. Объяснить такую популярность достаточно просто, технология не является особо энергозатратной, скорость в среде ее передачи достаточно высока. Однако технология достаточно стара и в большинстве устройств реализуется ее прямой приемник, имеющий гораздо больше возможностей. Этим приемником является технология NFC и судя по росту количества устройств, поддерживаемых эту технологию, темпы ее внедрения в различные механизмы будут только увеличиваться. Главным плюсом данной технологии является возможность организации соединения типа точка-точка, между двумя самостоятельными устройствами.

Целью данного дипломного проекта является реализация механизма аутентификации пользователя в операционной системе Windows средствами технологии NFC.

Задачи которые ставятся, это детальное рассмотрение предметной области и изучение специфики работы технологии NFC. Изучение методики организации связи между устройствами, поддерживающими данную технологию. И наконец, разработка программно-аппаратного комплекса, который будет удобен для пользователя и сможет реализовать поставленную цель проекта.

1 Исследование предметной области

1.1 Технология RFID

Радиочастотная идентификация (RFID) становится обычным явлением в повседневной жизни. От платежных карт «Онай» и транзитных пропусков до устройств E-ZPass, используемых на платных дорогах, наклеенных и прикрепленных к потребительским товарам бирок для предотвращения краж. Метка RFID состоит из небольшого радиоустройства; если быть более точным, радиоприемник и передатчик.

Существует две основные разновидности RFID: пассивные и активные метки. Процесс обмена данными в любой из разновидностей RFID задействует двух участников: цель и инициатор. Инициатор, устройство считывания тегов или устройство чтения / записи, запускает механизм обмена путем создания радиополя и прослушивания ответов от любой цели в поле. Цель или тег отвечает когда получает передачу от инициатора. Она ответит уникальным идентификатором (UID). RFID имеет два режима связи: активный и пассивный. При пассивном обмене RFID используются считыватель/записыватель и метка, на которой нет источника питания. Метки получают свою энергию от энергии самого радиополя. Как правило, это количество энергии невелико, однако, этого достаточно для того, чтобы отправить сигнал считывателю. Вариант с активной RFID-меткой включает цель, которая является устройством с независимым питанием. Поскольку цель включена, ее ответ читателю может пройти гораздо большее расстояние. E-ZPass и другие системы идентификации трафика используют активный RFID.

RFID-метки имеют небольшой объем памяти, обычно менее 1 килобайта. Устройство-инициатор может читать эти данные, и, если оно является устройством чтения/записи, оно также может записывать или изменять данные тега. Это позволяет хранить небольшие объемы информации. Например, иногда используется в транзитных системах, которые базируются на технологии RFID для отслеживания количества финансов накарте. Однако, поскольку системы RFID обычно объединены в сеть с базой данных, более распространенным является сохранение записи данных, индексированной в UID тега. В удаленной базе данных хранится вся информация связанная с индексом метки.

В последние несколько лет в связи с ростом популярности технологии RFID начал появляться новый термин: ближняя связь (NFC).

1.2 Технология NFC

NFC разработан на основе RFID, однако он позволяет обеспечить более сложные обмены данными между участниками. Плюсом данной технологии является возможность ограниченной обратной совместимости, если быть

более точным, технология поддерживает процесс чтения/записи пассивных RFID-меток. Помимо этого самым главным плюсом данной технологии является возможность организации двухстороннего или дуплексного режима передачи данных между устройствами.

Механизм работы NFC можно рассматривать как расширение возможностей RFID. Процесс обмена данными в NFC также состоит из инициатора и цели. Тем не менее, технология не ограничивается простым считыванием UID или записью небольшого объема данных на метку. Самое главное различие между RFID и NFC состоит в том, что цели NFC часто являются программируемыми устройствами, такими как смартфоны. Это означает, что вместо того, чтобы просто доставлять статические данные из памяти, цель NFC может фактически генерировать уникальный контент для каждого обмена и доставлять его обратно инициатору. Другими словами, передатчик осуществляя процесс передачи данных может получать ответ от устройства приемника о состоянии и корректности принятых данных.

Примерами таких радиопередач может быть быстрая синхронизация беспроводного стереофонического устройства, у которого, для передачи звукового файла используется Bluetooth, который имеет один минус, процесс сопряжения двух устройств занимает достаточно большое количество времени. Технология NFC позволяет совершить передачу пакетов на согласование подключения гораздо быстрее.

Главной особенностью процесса передачи данных по NFC является то, что во всем алгоритме действий вы не предоставляете второму устройству полный доступ к вашим данным, а лишь предоставляете все необходимые данные для совершения процесса сопряжения, при этом весь процесс полностью находится под вашим контролем.

Устройства NFC имеют два режима связи. Если инициатор всегда подает радиочастотную энергию, а цель получает питание от поля инициатора, говорят, что он работает в пассивном режиме связи. Если и цель, и инициатор имеют свои собственные источники энергии, они находятся в активном режиме связи. Эти режимы аналогичны обычным режимам связи RFID.

1.2.1 Стандартизация NFC

В пакет базовых стандартов NFC включены основные протоколы связи и вариации форматов обмена данными, а именно:

-ISO/IEC – 14443;

-FeliCa;

-MifareClassic;

-ISO/IEC – 18092;

Однако, помимо вышеперечисленных базовых стандартов существуют дополнения, которые были спроектированы организацией NFC Forum.

NFC Forum - это некоммерческая отраслевая ассоциация, созданная 18 марта 2004 года компаниями NXP Semiconductors, Sony и Nokia для продвижения использования беспроводного взаимодействия NFC в бытовой электронике, мобильных устройствах и персональных компьютерах. Стандарты включают в себя четыре различных типа тегов, которые совершают работу на различных скоростях и имеют различные возможности связи, охватывающие мультиплатформенность, объем памяти, безопасность хранения и процесса передачи, надежность сохранения данных и стойкость к различным воздействиям при записи. NFC Forum способствует внедрению и стандартизации технологии NFC для обеспечения взаимодействия устройств и сервисов.

1.2.2 Архитектура NFC

Для более четкого понимания механизма работы NFC рассмотрю модель архитектуры технологии. Она будет состоять из слоев, каждый из которых выполняет определенную роль. Самый нижний уровень - это физический, а именно клиентский процессор и радиостанции, которые осуществляют связь. В середине находятся уровни на которых данные представлены в виде пакетов, затем уровни конвертирования данных и, наконец, код приложения использующего технологию для работы. Более простым сравнение построенной схемы (рисунок 1.1) может служить модель OSI, которая осуществляет схожие процессы с данными с той лишь разницей, что применяется она в сетевых технологиях.

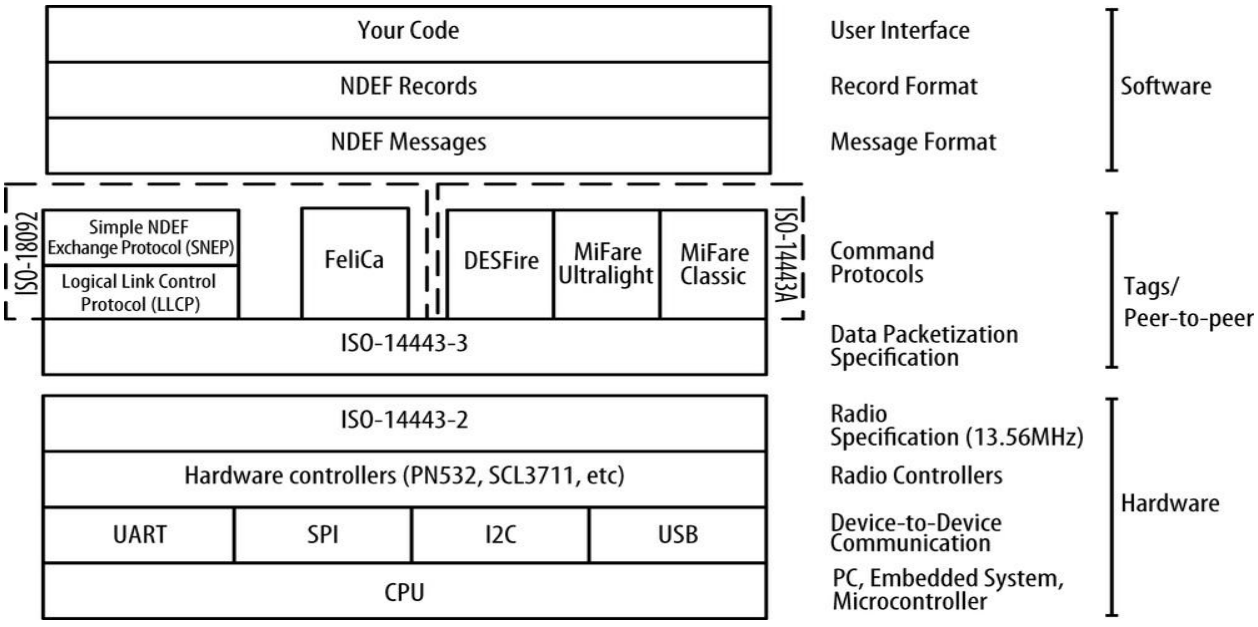

Рисунок 1.1 – Архитектура технологии NFC

На физическом уровне NFC работает над спецификацией радиочастот RFID ISO-14443-2, в которой описываются маломощные радиостанции,

работающие на частоте 13,56 МГц. Затем идет уровень, который описывает кадрирование байтов данных, передаваемых по радио, ISO-14443-3. Любые из радиостанций, которые могут использоваться, являются отдельными аппаратными компонентами, либо внутри телефона или планшета, либо в виде приложений для микроконтроллеров или персональных компьютеров. Они взаимодействуют с главным процессором устройства, используя один или несколько стандартных последовательных протоколов между устройствами: универсальный асинхронный протокол приема-передачи (UART), последовательный периферийный интерфейс (SPI), межинтегральная связь (I2C) или универсальная последовательная шина (USB). После двух уровней, основанных на спецификациях ISO идут несколько команд протокола RFID. Исходная спецификация управления RFID, на которой основано чтение и запись меток NFC - это ISO-14443A. Протоколы Philips / NXP Semiconductors Mifare Classic и Mifare Ultralight и NXP DESFire совместимы с ISO-14443A.

Одноранговый обмен NFC основан на протоколе управления ISO-18092. RFID-карты и метки Sony FeliCa, применяемые в японской промышленности, также основаны на этом стандарте.

Первым отличием технологии NFC от RFID является режим одноранговой связи, который реализован с использованием стандарта ISO-18092. Существует два протокола: протокол управления логическим каналом (LLCP) и простой протокол обмена NDEF (SNEP), которые управляют одноранговыми обменами. Они отображаются в стеке вместе с протоколами управления, поскольку они работают по тому же стандарту.

Второе основное различие между RFID и NFC, формат обмена данными NFC (NDEF). NDEF определяет обмен данными в сообщениях, которые состоят из записей NDEF.

1.2.3 Типы NFC-меток

Существует четыре типа меток, определенных форумом NFC и все они основаны на протоколах управления RFID. Помимо четырех основных, существует еще пятый тип, который совместим, но не является строго частью спецификации NFC. Типы 1, 2 и 4 основаны на ISO-14443A, а тип 3 основан на ISO-18092. Разберем каждый из типов тегов:

Первый тип: основан на спецификации ISO-14443A. Может быть только для чтения или для чтения / записи. Имеет от 96 байт до 2 килобайт памяти. Скорость связи 106Кб. Не имеет защиты от коллизий данных. Пример: Broadcom BCM20203;

Второй тип: подобно меткам типа 1, метки типа 2 основаны на спецификации NXP / Philips Mifare Ultralight (ISO-14443A). Может быть только для чтения или для чтения / записи. Имеет от 96 байт до 2 килобайт памяти. Скорость связи аналогична первому типу. Присутствует антиколлизионная поддержка. Пример: NXP Mifare Сверхлегкий;

Третий тип: основан на тегах Sony FeliCa (ISO-18092 и JIS-X-6319-4), без поддержки шифрования и аутентификации, которую предоставляет FeliCa. Конфигурируется на заводе-изготовителе только для чтения или с возможностью чтения / записи. Имеет переменную память в размере до 1 МБ. Поддерживает два типа скоростей работы, 212 или 424 Кбит/с. Присутствует антиколлизионная поддержка. Пример: Sony FeliCa;

Четвертый тип: аналогично тегам первого типа, основаны на спецификации тега NXP DESFire (ISO-14443A). Конфигурируется на заводе-изготовителе только для чтения или с возможностью чтения/записи. Имеют 2, 4 или 8 КБ памяти. Поддерживают три скорости связи: 106, 212 или 424 Кбит/с. Имеют антиколлизионную поддержку. Примеры: NXP DESFire, SmartMX-JCOP.

1.2.4 Формат обмена данными - NDEF

NDEF - это стандартизированный формат данных, который можно использовать для обмена информацией между любым совместимым устройством NFC и другим устройством или тегом NFC. Формат данных состоит из сообщений (потока данных, которым обмениваются устройства) NDEF и записей (меток или тегов) NDEF. Стандарт поддерживается NFC Forum и находится в свободном доступе для консультаций, но для его загрузки необходимо принять лицензионное соглашение .

Формат NDEF используется для хранения и обмена информацией, такой как URI, простой текст. С использованием общепринятого формата. Теги NFC, такие как карты Mifare Classic, можно настроить как теги NDEF, а данные, записанные в них одним устройством NFC (записи NDEF), могут быть поняты и доступны любому другому устройству, совместимому с NDEF. Сообщения NDEF также могут использоваться для обмена данными между двумя активными устройствами NFC в режиме «peer-to-peer».

Транзакции NFC, как правило, короткие. Каждый обмен обычно состоит только из одного сообщения, и каждый тег содержит только одно сообщение. Исходя из описанного можно привести следующий пример, предположим, вы захотите передать всю информацио о вашей организации вашему компаньону, для этого вам придется разбить информацию на части, поскольку передать большой объем данных мы не можем. Значит, мы разбиваем всю информацию на части, как показано на рисунке 1.2.

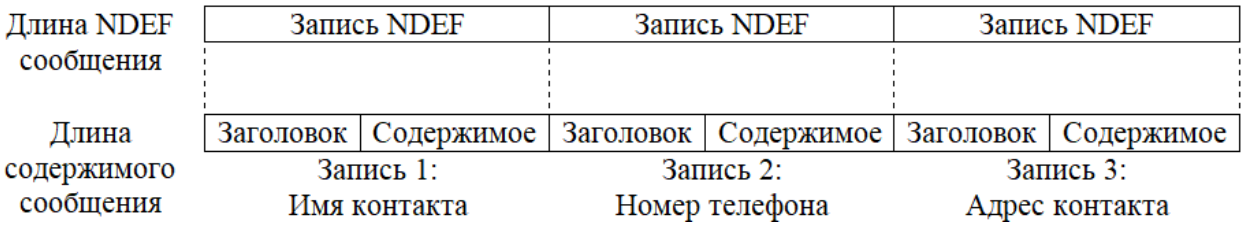

Рисунок 1.2 – Общая структура сообщения NDEF

На рисунке 1.2 показан лишь общий пример содержимого каждого из сообщений, перейду к более детальному разбору одного сообщения. NDEF запись содержит полезные данные и метаданные, описывающие, как интерпретировать информацию, содержащуюся в части полезно нагруженного сообщения. Полезная нагрузка каждой записи может быть одного из нескольких типов данных. Заголовок для каждой записи содержит метаданные, описывающие запись и ее место в сообщении, а также ее тип и идентификатор. После заголовка идет полезная нагрузка. Рисунок 1.3 дает полное представление о битах и байтах записи NDEF.

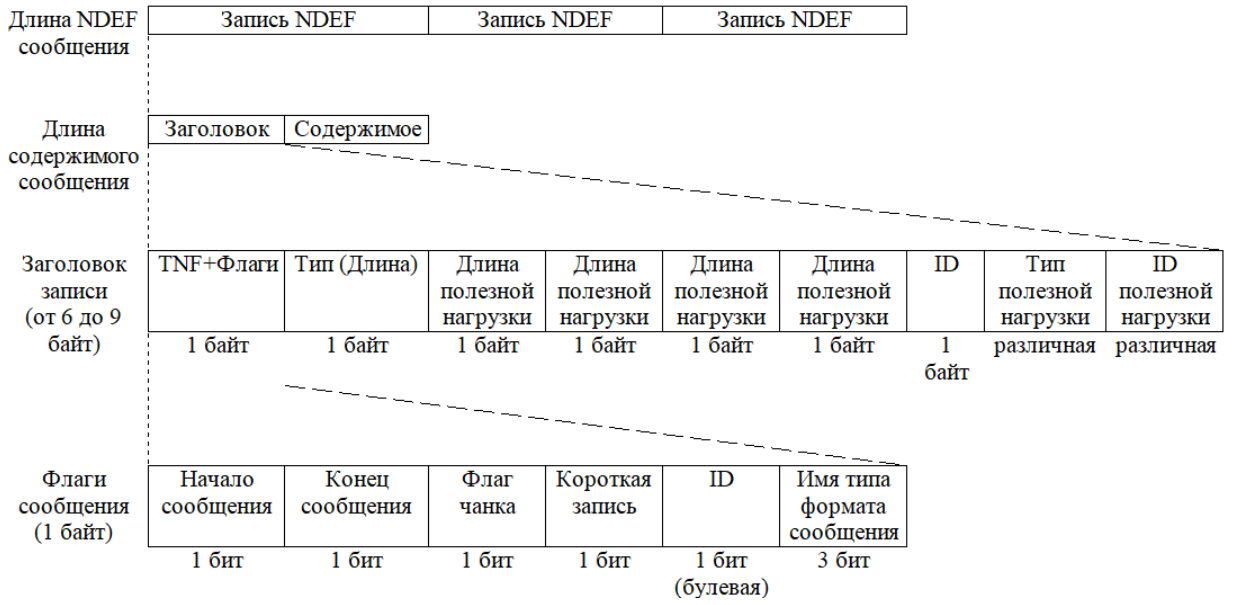

Рисунок 1.3 – Детальная структура NDEF сообщения

Из рисунка 1.3 видно, запись NDEF состоит из формата имени типа (TNF), типа полезной нагрузки, идентификатора полезной нагрузки и полезной нагрузки. Полезная нагрузка является наиболее важной частью записи NDEF; это контент, который вы передаете. TNF говорит как интерпретировать тип полезной нагрузки. Тип полезной нагрузки - это тип, специфичный для NFC, медиа-тип MIME или URI, который говорит, как интерпретировать полезную нагрузку. Иначе говоря TNF - это метаданные о типе полезной нагрузки, а тип полезной нагрузки - это метаданные о полезной нагрузке. Идентификатор полезной нагрузки является

необязательным и позволяет комбинировать полезные нагрузки или создавать пересеющиеся сооющения.

Существует семь возможных значений TNF:

0 Пусто - пустая запись, которая не имеет типа или полезной нагрузки.

1 Хорошо известная (англ. Well-Known) - один из нескольких предопределенных типов, изложенных в спецификации RTF NFC Forum.

2 MIME - тип мультимедиа данных, определенный в RFC 2046.

3 URI - ссылка на ресурс, как определено в RFC 3986.

4 Внешний (англ. External) - пользовательское значение, основанное на правилах в спецификации определения типа записи форума NFC.

5 Неизвестный - длина данных такого типа равна 0.

6 без изменений - только для средних и конечных записей фрагментированных полезных нагрузок.

7 Зарезервировано - зарезервированный форумом NFC для будущего использования.

Идентификатор полезной нагрузки, который является необязательным полем, должен быть действительным URI. Он используется, чтобы позволить вашим приложениям идентифицировать полезную нагрузку в записи по идентификатору или для создания цепочек полезных нагрузок при мередаче нескольких сообщений.

Полезная нагрузка - это содержимое передаваемого сообщения. Это может быть что угодно, что умещается в поток байтов. Правильно построенная библиотека NDEF не заботится о том, что находится в полезной нагрузке, она просто передает ее. Полезная нагрузка в содержимом сообщения может быть зашифрована при передаче, и на уровне приложения снова дешифрована для ее изучения. Отправляющее и принимающее приложения должны согласовать, что означает полезная нагрузка и как она отформатирована.

Разберу детально последнюю строку рисунка 2, которая обозначается как флаги сообщения. Первые пять битов записи NDEF - это флаги, которые указывают, как обрабатывать запись, и информацию о местонахождении записи в сообщении.

Битовые флаги в первом байте заголовка записи следующие:

Начало сообщения - выставлен, когда это первая запись в сообщении;

Конец сообщения - выставлен, когда это последняя запись в сообщении;

Флаг чанка – выставлен, когда эта запись фрагментирована;

Короткая запись – выставлен, если для длины полезной нагрузки используется формат короткой записи;

ID – выставлен, если поле длины идентификатора присутствует.

Обращаю внимание, что бит IL не является длиной поля идентификатора, он просто указывает на наличие поля длины. Если бит IL равен 0, поле длины идентификатора или поле идентификатора отсутствует.

Битовые флаги «Начало сообщения» и «Конец сообщения» используются для обработки записей в сообщении. Поскольку сообщение

NDEF представляет собой просто набор из одной или нескольких записей NDEF, для сообщения NDEF нет двоичного формата. Флаги «Начало сообщения» и «Конец сообщения» позволяют вам определить, когда сообщение начинается и заканчивается. Первая запись в сообщении будет иметь флаг «Начало сообщения» установленный в true. Для средних записей оба флага будут установлены в false. Конечная запись в сообщении будет иметь значение «Конец сообщения». Для сообщения с одной записью биты начала сообщения и конца сообщения будут иметь значение true.

1.2.4.1 Физические возможности формата NDEF

Размер полезной нагрузки записи NDEF ограничен 2^{32}–1 байтов, поэтому поле длины полезной нагрузки заголовка составляет четыре байта (или 2^{32} бита). Однако записи могут быть объединены в сообщение для формирования более длинных полезных нагрузок. В теории нет ограничений на длину сообщения NDEF. На практике возможности устройств и тегов определяют пределы. Если обмен происходит одноранговыми сообщениями между устройствами и теги не используются, сообщения NDEF ограничиваются только вычислительной мощностью устройств и терпением человека, удерживающего два устройства в поле действия антены. Однако если обмен данными происходит между устройством и тегом, сообщения ограничены объемом памяти тега.

Когда используются метки NFC, ограничения на размер записей значительно ниже 2^{32}-1 байт. Типы тегов NFC основаны на нескольких различных стандартах RFID. Большинство типов меток NFC основаны на стандарте ISO-14443A. Они варьируются от 96 байт памяти с возможностью расширения до 4 КБ в зависимости от типа тега. Теги семейства Philips / NXP Mifare совместимы с NFC, в том числе сверхлегкие метки Mifare, метки Mifare Classic 1K и 4K и Classic Mini. Существует один тип тегов NFC, основанный на японском промышленном стандарте (JIS) X 6319-4. Они имеют до 1 МБ памяти. Теги Sony FeliCa являются типичными представителями тегов такой технологии.

Как правило, обмены данными в NFC короткие. Человек подносит свое устройство к тегу или другому устройству, происходит краткий обмен, и на этом процесс передачи заканчивается.

Вывод

В данной главе были рассмотрены технологии беспроводной передачи данных RFID и NFC, детально изучена специфика работы механизма отправки сообщений по каждому из уровней, на которых происходит работа с данными. Расписан стандарт отправки сообщений NDEF, а именно структура пакетов данного стандарта, возможные варианты отправки информации различных типов. Помимо вышеперечисленного был изучен альтернативный метод аутентификации пользователей в операционной системе семейства Windows.

Исходя из изученного материала представляется разумным разработать программно-аппаратный комплекс, базирующийся на технологии беспроводной передачи NFC, который бы позволил реализовать безопасный доступ к учетной записи системы.

Благодаря тому, что Windows cridential provider поддерживает возможность разработки собственного механизма аутентификации пользователей в системе, я высказал предположение, о том, что можно реализовать взаимодействие со сторонним устройством, которое будет отправлять данные полученные при помощи технологии NFC. Данным сторонним устройством может быть отдельно подключаемый модуль или уже встроенный в рабочую систему.

2 Инструменты и технологии проекта

2.1 Среды разработки

Для реализации поставленной задачи дипломного проекта как программной, так и аппаратной частей были задействованы следующие инструменты:

- Microsoft Visual Studio 2017 (C#);
- Androud Studio (Java);
- Arduino IDE (C);
- EasyEDA.

I. Visual Studio используется для разработки программного обеспечения под операционную систему Windows. Выбор пал на Visual Studio ввиду достаточно простого, локоничного и удобного интерфейса разработки Windows Forms типа программ. Среда разработки позволяет опустить тонкости настройки конфигурации форм и визуальных объектов приложения делая это автоматически. Благодаря этому появляется больше времени на написание основонго функционального кода. Также, существует достаточно обширное количество литературы, описывающее различные методики разработки и решения возникающих проблем. На рисунке 2.1 приведен скрин окна, в котором и происходит работа над проектом.

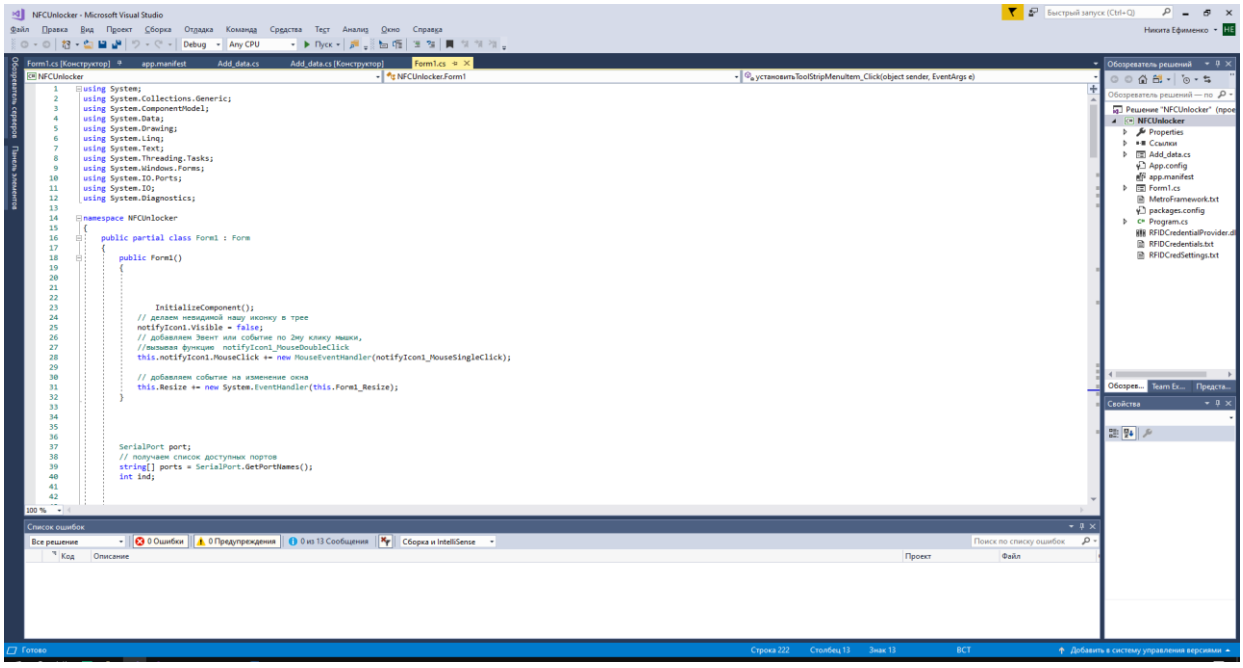

Рисунок 2.1 – Окно среды разработки Microsoft Visual Studio 2017

Помимо вышеописанных возможностей среды разработки стоит отметить ее мультиязычность, что означает, что в ее базовый комплект поддерживаемых языков программирования входят C, C++, C#, Java, Python и многие другие. Для выполнения данного проекта использовался C#.

Обосновывается это личными навыками автора, а также наличием обширного количества хорошо написанных библиотек и нацеленность самого языка на объект-ориентированное программирование.

II. Для разработки программного обеспечения под операционную систему Android была выбрана среда разработки Android Studio. Благодаря тому, что она является официально интегрированной для Google, процесс разработки и тестирования протекает довольно легко не только на программных эмуляторах, но и на реальном железе. Android Studio поддерживает двая языка программирования: Java и Kotlin. Благодаря достаточно интересному механизму отладки проекта, процесс сборки не занимает большого количества времени. При изначально запущенном программном эмулятора, после сборки уже можно визуально увидеть полученный результат написанного кода в действии (рисунок 2.2).

При написании проекта в данной среде разработки был выбран язык программирования Java. Выбор можно обосновать достоаточно длительным временем существования самого языка, что позволяет сэкономить время при возникновении частовстречаемых ошибок при написании кода или его компиляции и сборке.

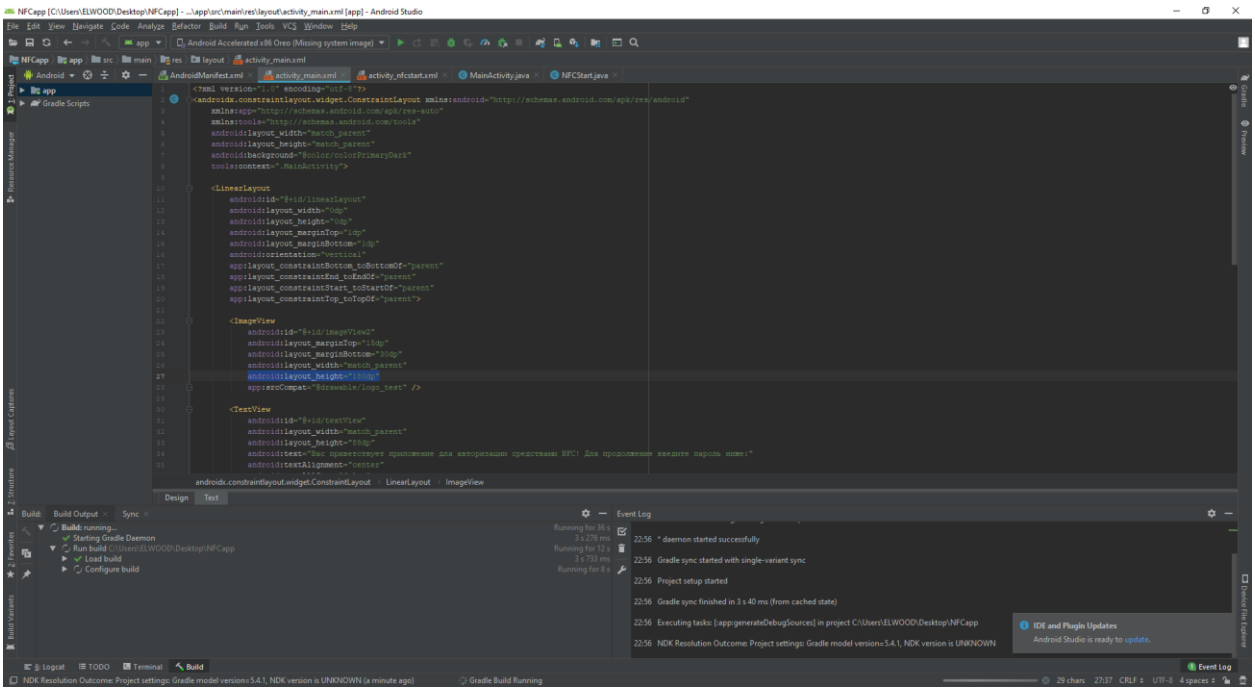

Рисунок 2.2 - Окно среды разработки Android Studio

III. Помимо написания программных элементов проекта, требуется создать устройство, которое будет поддерживать технологию NFC и при этом позволит совместить 2 программы разработанные в вышеописанных средах. Для решения данного вопроса был выбран инструмент Arduino IDE. Он представляет из себя программное обеспепечение от официальных разработчиков Arduino с открытым исходным кодом. Основной задачей является компиляция проекта и последующая запись на плату

микроконтроллера. Написание программного кода происходит на языке программирования C++ (рисунок 2.3).

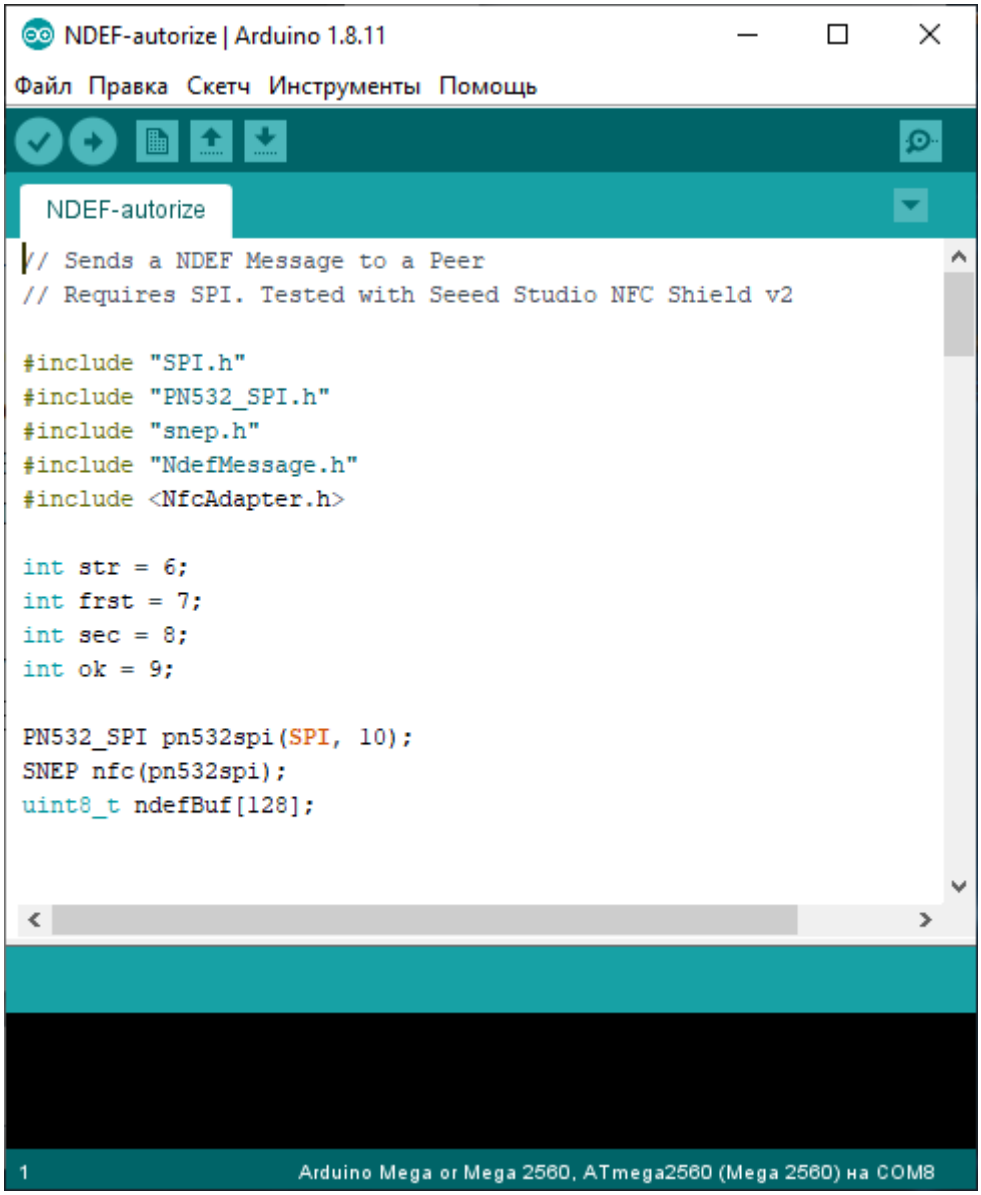

Рисунок 2.3 – Рабочее окно программы Arduino IDE

IV. Последний из списка используемых инструментов - EasyEDA. Программное обеспечение, позволяющее осуществить процесс проектирования расположения компонентов устройства на плате, произвести корректное размещение дорожек, рассчитать правильные параметры компонентов,а также по итогам завершения процесса проектирования произвести заказ на изготовление платы. Плюсами данного программного обеспечения является его кросплатформенность с поддержкой даже веб-интерфейса и возможность облачного хранения базы проектов (рисунок 2.4).

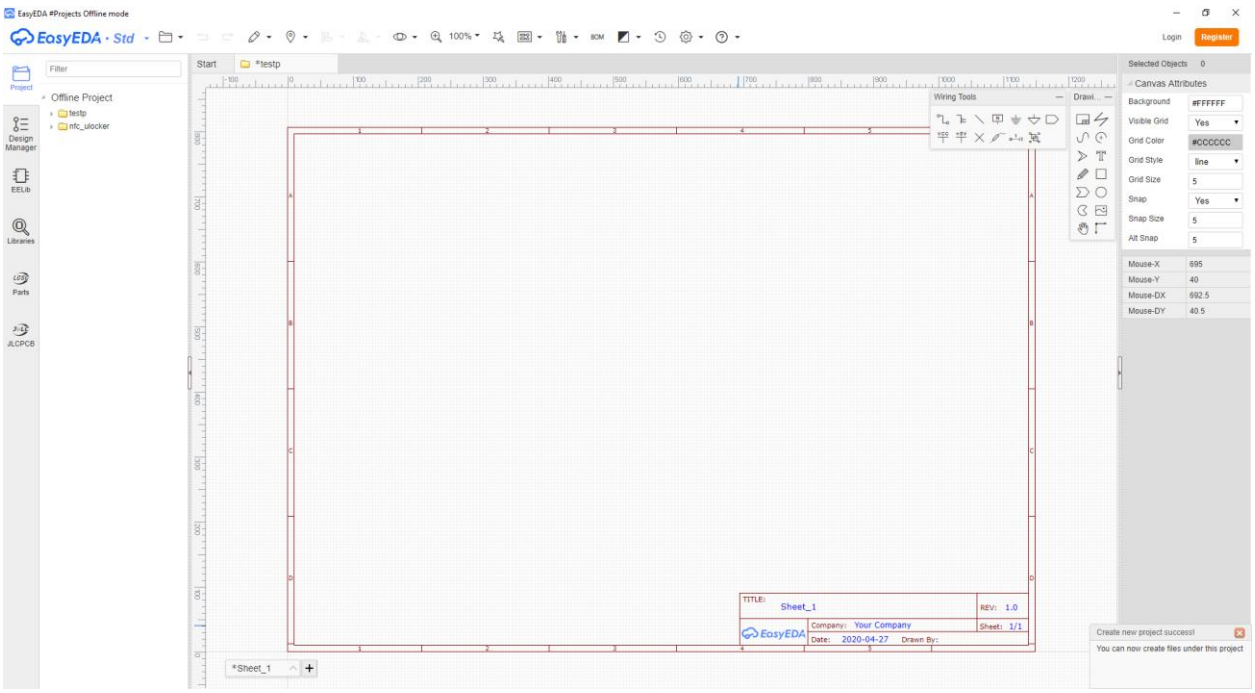

Рисунок 2.4 - Рабочее окно программы EasyEDA

2.2 Windows Credential Provider

WCP (Windows Credential Provider) – это специальный компонент операционной системы Windows, который осуществляет собственный способ получения пользовательских учетных данных для входа в систему. В некоторых источниках данная технология обозначается как: «Поставщики учетных данных». В базовый комплект Microsoft входит множество поставщиков учетных данных в составе Windows, таких как пароль, PIN-код, смарт-карта и Windows Hello (распознавание отпечатков пальцев, лиц и Iris).

Среда поставщика учетных данных Windows позволяет разработчикам создавать собственные поставщики учетных данных. Когда Winlogon хочет собрать учетные данные, пользовательский интерфейс входа в систему запрашивает у каждого поставщика учетных данных количество известных им учетных данных. После того, как все провайдеры перечислили свои списки, пользовательский интерфейс входа отображает их для пользователя. Затем пользователь взаимодействует с плиткой для предоставления необходимых учетных данных. Пользовательский интерфейс входа в систему передает эти учетные данные для проверки подлинности.

Благодаря этой системе намного проще создать провайдер учетных данных, чем это было ранее. Большая часть работы выполняется комбинацией Winlogon, пользовательского интерфейса Logon и пользовательского интерфейса Credential. Для реализации механизма необходимо создать собственную реализацию ICredentialProvider и ICredentialProviderCredential.

Важно отметить, что поставщики учетных данных не являются правоприменительными механизмами. Они просто используются для сбора и сериализации учетных данных, представляя их для авторизации. Пакеты локальных полномочий и аутентификации будут обрабатывать все необходимые меры безопасности.

Комбинируя провайдеров учетных данных с поддерживаемым оборудованием, возможно расширить Windows для поддержки входа с биометрической информацией, паролями, ПИН-кодами, сертификатами смарт-карт или любым пользовательским пакетом аутентификации. Также можно настроить пользовательский интерфейс для входа в систему различными способами. Например, когда пользовательский интерфейс входа в систему запрашивает у вашего провайдера учетных данных листы учетных данных, возможно указать плитку по умолчанию, чтобы предоставить пользователю настраиваемый интерфейс. Поставщики учетных данных могут даже быть рассчитаны на поддержку единого входа (SSO), аутентификации пользователей в защищенной точке доступа и входа в систему с компьютера.

2.3 Android Beam(NDEF)

Считывание данных NDEF из тега NFC обрабатывается системой диспетчеризации тегов, которая анализирует обнаруженные теги NFC, соответствующим образом классифицирует данные и запускает приложение, которое заинтересовано в категорированных данных. Приложение, которое хочет обработать отсканированный тег NFC, может объявить фильтр намерений и запросить обработку данных.

Функция Android Beam позволяет устройству передавать сообщение NDEF на другое устройство. Это взаимодействие обеспечивает более простой способ отправки данных, чем другие беспроводные технологии, такие как Bluetooth, поскольку при использовании NFC не требуется обнаружение или сопряжение устройства вручную. Соединение автоматически запускается, когда в зону действия попадают два устройства. Android Beam доступен через набор API NFC, поэтому любое приложение может передавать информацию между устройствами. Например, приложения «Контакты», «Браузер» и «YouTube» используют Android Beam для соответствующего обмена контактами, веб-страницами и видео с другими устройствами.

Основной способ работы системы диспетчеризации тегов заключается в следующем:

1. Запуск приложения с параметрами, которые были сгенерированы системой диспетчеризации тегов при контакте с устройством (либо, ACTION_NDEF_DISCOVERED либо ACTION_TECH_DISCOVERED).

2. Если никакие действия не происходят при контакте с устройством, происходит переход на запуск с более низкими приоритетами (или, ACTION_TECH_DISCOVEREDили ACTION_TAG_DISCOVERED), пока

приложение не отфильтрует намерение или пока система диспетчеризации тегов не попробует все возможные намерения.

3. Если ни одно из приложений не отфильтровывает ни одно из намерений, никакой реакци не происходит.

Весь вышеописанный способ можно представить в графическом виде на рисунке 2.5.

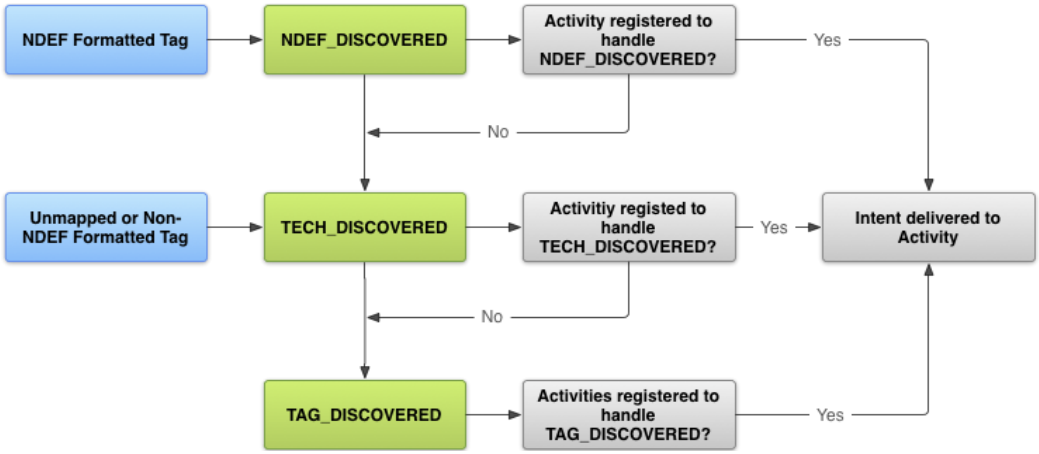

Рисунок 2.5 – Схема реакции системы диспетчеризации тегов

2.4 Arduino NDEF

В дипломном проекте для реализации задачи по созданию устройства, связующего персональный компьютер и смартфон средствами NFC используется микроконтроллер на базе Arduino. Выбор обусловлен тем, что программное и аппаратное обеспечение Arduino имеют открытый исходный код. В основе обеспечения работоспособности NFC в Arduino лежит радиоконтроллер. Одним из самых популярных контроллеров NFC на рынке сейчас является NXP PN532. PN53x можно найти в большинстве потребительских устройств с поддержкой NFC на рынке. Существует несколько модулей, совместимых с Arduino, которые используют PN532: отладочная плата контроллера NFC / RFID PN532 v1.3 от Adafruit; плата контроллера NFC / RFID PN532 для Arduino (также от Adafruit); и Seeed Studio NFC Shield и NFC Shield v2.0. В данном проекте таким радиомодулем является недорогой аналог вышеперечисленных плат, а именно NFC module v3 от компании Elechouse. На рисунке 2.6 показан вид внешнего вида радиомодуля.

PN532 NFC модуль имеет в своем распоряжении три различных интерфейса связи с микроконтроллером. Он может общаться с использованием асинхронной последовательной связи, с использованием универсального асинхронного приемника-передатчика (UART). Также может использовать две формы синхронной последовательной связи: последовательный периферийный интерфейс (SPI) или межинтегральная

связь (I2C). Асинхронная последовательная связь обычно используется между двумя компьютерами, каждый из которых имеет свои собственные внутренние часы и может работать независимо, в то время как SPI и I2C используются для связи между главным компьютером и устройствами. Асинхронный последовательный интерфейс всегда является связью один-к-одному, но I2C и SPI являются протоколами шины, это означает, что вы можете иметь несколько индивидуально адресуемых устройств, использующих одни и те же линии связи.

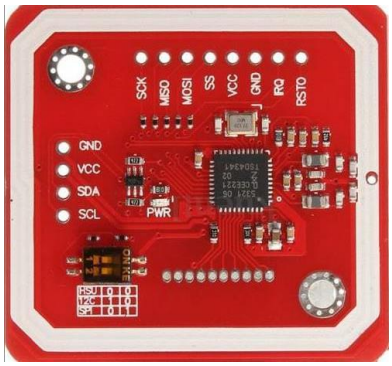

Рисунок 2.6 – Радиомодуль PN532

Библиотеки радиомодуля PN532 состоят из низкоуровневых команд для создания связки сикроконтроллер-радиомодуль. Поддерживают I2C и SPI режимы. Сами по себе они просто доставляют данные из тегов в виде строки байтов. Чтобы интерпретировать эти байты, требуются некоторые вспомогательные библиотеки, которые описывают различные протоколы тегов: Mifare Classic, Mifare Ultralight и т. д. Наконец, кроме того, есть библиотека NDEF, которая считывает байты, переданные из библиотек типов тегов, как последовательность сообщений и записей NDEF. Без библиотеки NDEF пришлось бы разбираться побайтно, что происходит с байтами, поступающими из тегов. С её помощью возможно вообще игнорировать то, что происходит на более низких уровнях, и сосредоточиться на чтении и написании сообщений NDEF. На рисунке 2.7 показан стек библиотек и связь между ними во время работы микроконтроллера и радиомодуля.

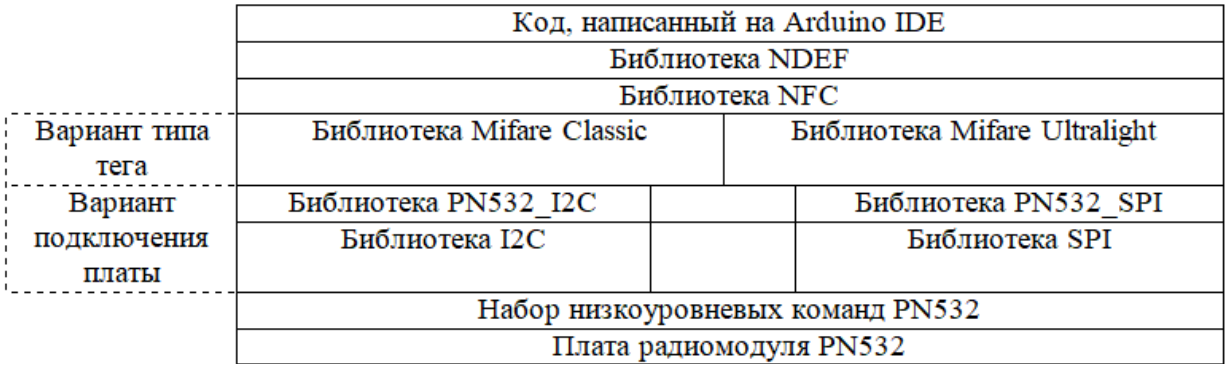

Рисунок 2.7 – Схема связей библиотек

Вывод

В данной главе были рассмотрены основные инструменты, симбиоз которых должен позволить реализовать поставленную цель перед дипломным проектом. Для решения задачи по разработке программных средств были выбраны наиболее оптимальные и свежие среды разработки с достаточно обширной базой информации, так или иначе касающейся специфики дипломного проекта. Помимо программного обеспечения, специфика диплома подразумевает разработку аппаратной части. Инструменты, используемые для решения данной задачи были определены исходя из степени открытости исходного кода библиотек, применяемых при разработке, а также доступности ресурсов для студента. Помимо сред разработки были рассмотрены технологии, которые могут обеспечить совместную работу разработанных программ и устройства.

Благодаря тому, что Windows cridential provider поддерживает возможность разработки собственного механизма аутентификации пользователей в системе, можно реализовать взаимодействие со сторонним устройством, которое будет отправлять данные полученные при помощи технологии NFC. Данным сторонним устройством может быть отдельно подключаемый модуль или уже встроенный в рабочую систему.

Решением вопроса передачи данных на устройство стало использование технологии Android Beam (NDEF). Поскольку технология входит в пакет базовых API системы Android, обычному пользователю не требуется наличие привилегированных прав для работы с разработанным приложением. Это позволяет упростить механизм взаимодействия пользователя с приложением, а так расширить список поддерживаемых смартфонов ввиду отсутствия надобности получения root доступа.

Благодаря разработанной энтузиастами библиотеке NDEF для модуля PN532, появилось больше времени сосредоточится на решении вопроса декодирования полученного сообщения и передаче его в операционную систему Windows.

3 Практическая часть

3.1 Устройство на Arduino

Одним из важнейших пунктов дипломного проекта является разработка устройства, которое обеспечит возможность передачи данных между компьютером и смартфоном посредством NFC. Для реализации была выбрана платформа Arduino и модуль PN532.

3.1.1 Характеристики устройства

Приведу элементы в удобном виде таблицы 3.1. Разберу характеристики каждого элемента в частности (таблицы 3.2, 3.3, 3.4).

Таблица 3.1 – Элементы устройства

Наименование	Кол-во (шт)	Описание
Микроконтроллер Arduino NANO	1	для управления радиомодулем и отправки данных на COM-порт
Радиомодуль PN532	1	для отправки/принятия данных по NFC
Светодиод красный	3	для индикации статуса устройства
Светодиод зеленый	1	
Резистор	4	для корректной работы светодиодов

Таблица 3.2 – Характеристики микроконтроллера

Микроконтроллер	ATmega328
Рабочее напряжение	5 В
Входное напряжение (рекомендуемое)	7-12 В
Входное напряжение (предельное)	6-20 В
Цифровые входы/выходы	14 (6 могут как выходы ШИМ)
Аналоговые входы	8
Постоянный ток через вход/выход	40 мА
Флеш-память	32 Кб (ATmega328), 2 Кб для загрузчика
ОЗУ	2 Кб (ATmega328)
EEPROM	1 Кб (ATmega328)
Тактовая частота	16 МГц
Размеры	1.85 см x 4.2 см

Таблица 3.3 – Характеристики радиомодуля

Контроллер	NXP PN532
Режимы работы	I2C, SPI и HSU
Поддерживаемые стандарты	Карты Mifare 1k, 4k, Ultralight и DesFire; карты ISO / IEC 14443-4, такие как CD97BX, CD light, Desfire, P5CN072 (SMX); карты Innovision Jewel, такие как карта IRT5001; карты FeliCa, такие как RCS_860 и RCS_854
Питание	5 В TTL для I2C и UART, 3.3 В TTL SPI

Таблица 3.4 – Характеристики светодиодов и сопротивления

Наименование	Характеристика
Светодиод красный	Падение напряжения 1,5 В
Светодиод зеленый	Падение напряжения 1,7 В
Сопротивление	220 Ом

3.1.2 Схема подключения

Исходя из элементов и их характеристик (указанных в пункте 3.1.1) нужно произвести подключение и разводку платы для увеличения компактности устройства. Используя программу EasyEDA произвожу виртуальную сборку платы и разметщаю расположение дорожек. На рисунке 3.1 показан результат выполненной работы. Сборка схемы производилась на макетной плате размерностью 60x70 мм.

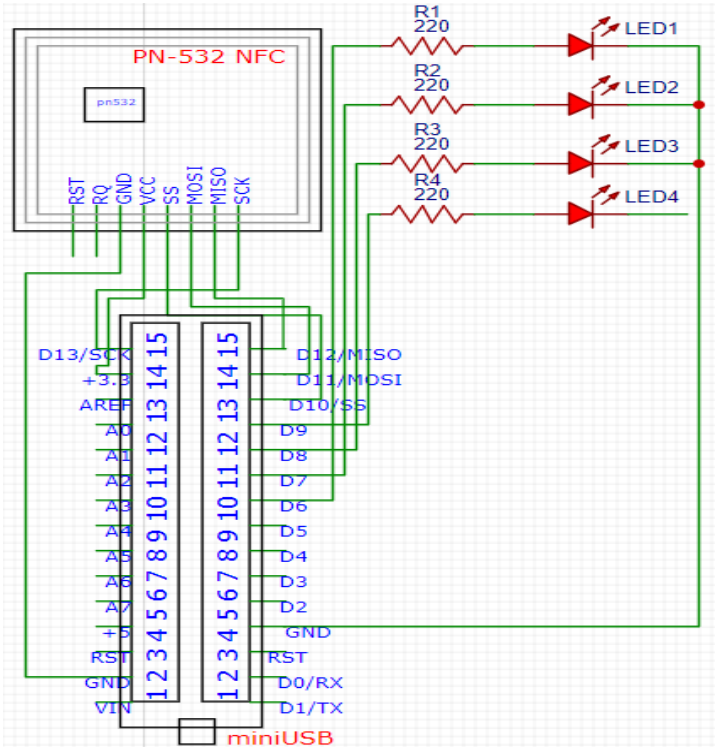

Рисунок 3.1 – Схема устройства

3.1.3 Написание прошивки

После завершения этапа проектирования платы и её сборки можно приступить к написанию прошивки для устройства. Разработаю схему алгоритма работы программы, которая сократит время написания программного кода (рисунок 3.2).

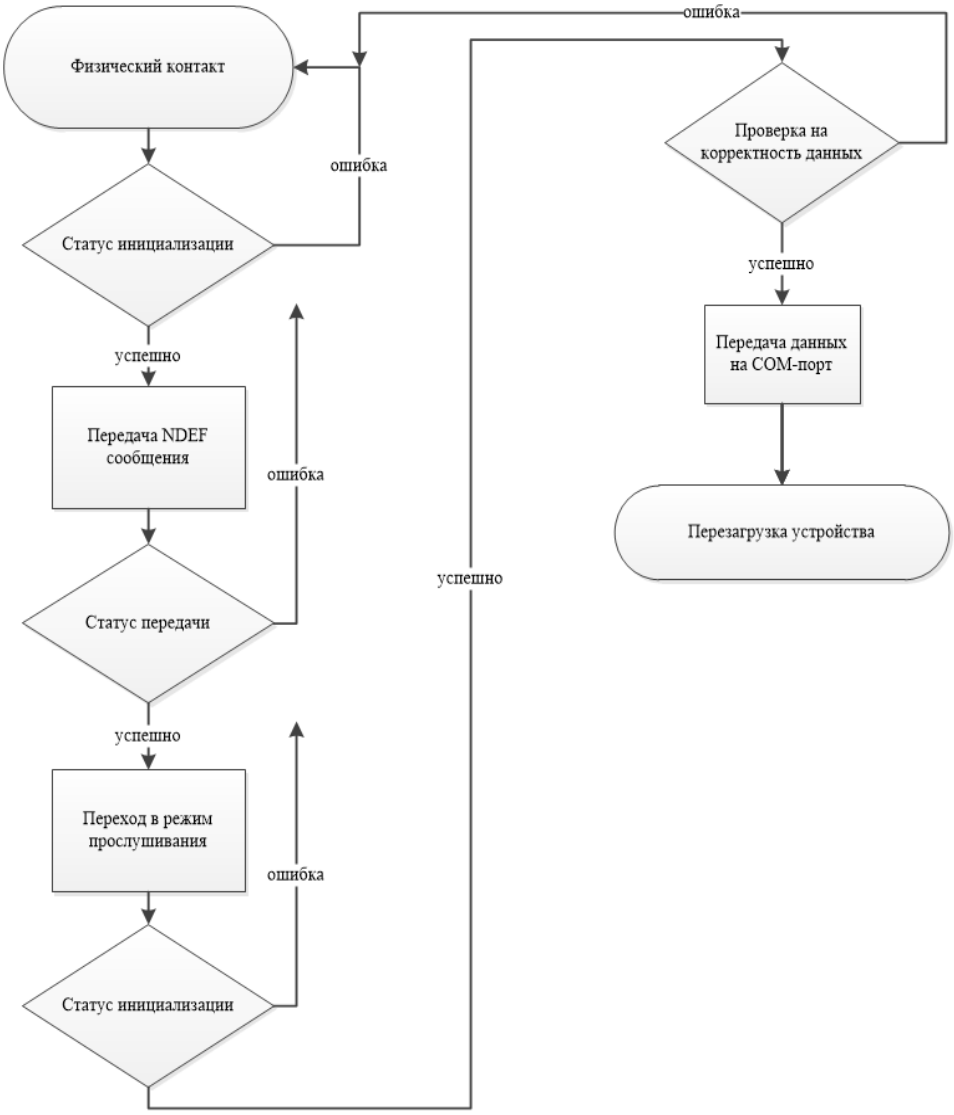

Рисунок 3.2 – Блок-схема работы алгоритма прошивки

Благодаря разработанной блок-схеме заметно, что тип алгоритма решения поставленной задачи является ветвящимся. Следовательно, в коде должны использоваться логические операторы. Используя среду разработки Arduino IDE и примеры работы с радиомодулем PN532, пишу программный код. На рисунках 3.3, 3.4 показаны основные моменты кода, которым уделялось большое внимание при его написании.

```
//Задание переменной message, хранящей NDEF сообщение
NdefMessage message = NdefMessage();
message.addTextRecord("Start te Ulocker!");
//проверка на корректность установленного значения переменной message
int messageSize = message.getEncodedSize();
if (messageSize > sizeof(ndefBuf)) {
    leder();
    //Запуск авторитма трансляции сообщения на смартфон
    while (1) {
    }
}
//проверка на корректность установленного соединения
message.encode(ndefBuf);
if (0 >= nfc.write(ndefBuf, messageSize)) {
    //Индикация ошибки
    leder();
} else {
    //Индикация перехода на второй этап
    ledok();
```

Рисунок 3.3 – Режим транлсяции сообщения от устройства

```
//Формирования массива данных полученного сообщения
NdefMessage msg  = NdefMessage(ndefBuf, msgSize);

//вывод части payload из сообщения ndef
NdefRecord record = msg.getRecord(0);
int payloadLength = record.getPayloadLength();
byte payload[payloadLength];
record.getPayload(payload);
String payloadAsString = "";
//Посимвольная запись части массива, содержащей кодовую комбинацию
for (int c = 0; c < payloadLength; c++)
{
  payloadAsString += (char)payload[c];
}
```

Рисунок 3.4 – Работа с полученными данными

В приложении А приведен весь код прошивки.
При написании прошивки устройства были использованы библиотеки:

- include "SPI.h";
- include "PN532_SPI.h";
- include "snep.h";
- include "NdefMessage.h";
- include <NfcAdapter.h>.

3.1.4 Описание устройства

Для облегчения идентификации устройства, ему было присвоено наименование «НТМА v1.0». Оно представляет из себя коробочку с размерами 69,4 х 53,6 х 23,7 мм. На рисунках 3.5, 3.6, 3.7 приведены снимки внешнего вида устройства.

Помимо указания наименования устройства присутствует световая индикация работы устройства с текстовым расшифрованием каждого из зажигаемых светодиодов. Верхний светодиод питания светится постоянно, поскольку характеризует наличие питания на устройстве. Вторым после питания идет светодиод, индикация которого характеризует режим перехода устройства из стадии отправки сообщения от устройства к смартфону в стадию приема сообщения от смартфона на устройства. Ниже расположены 2 светодиода, отвечающие за результаты каждой из происходимых операций на устройстве, соответственно: зеленый светодиод отвечает за успешное завершение работы, а красный за присутствие ошибок.

Рисунок 3.5 – Лицевая сторона устройства

Рисунок 3.6 – Боковая сторона устройства

Для выполнения процесса прошивки или обеспечения передачи данных между устройством и компьютером используется порт miniUSB, расположенный на правой боковой стороне устройства.

Внутренняя компоновка устройства представляет из себя макетную плату, на которой произведено размещение компонентов (рисунки 3.7-3.8). Рисунки 3.9 и 3.10 отображают расположение и технологию подключения.

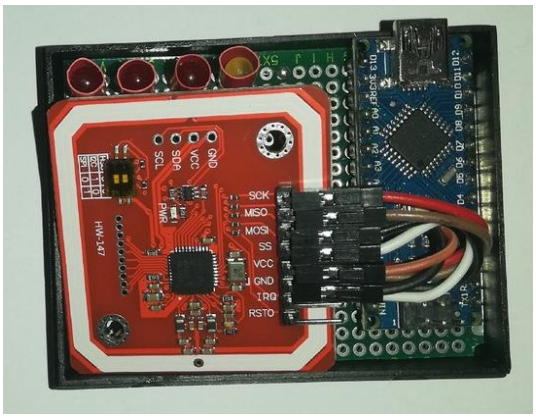

Рисунок 3.7 – Внутреняя компановка устройства (вид сверху)

Рисунок 3.8 – Внутреняя компановка устройства (вид сбоку)

Как видно из рисунка 3.8 модуль PN532 подключен посредством быстросъемных коннекторов, присутствует свободный ход модуля относительно макетной платы. Это сделано для сокращения расстояния между платой модуля и лицевой крышкой устройства. Поскольку антена не является отдельным, подключаемым модулем, требуется как можно более тесное соприкосновение смартфона с радиомодулем.

Для решения проблемы засвечивания индикаторов, на их корпус надеты небольшие отрезки термоусадки. Что позволяет более точно определить какой из светодиодов загорается.

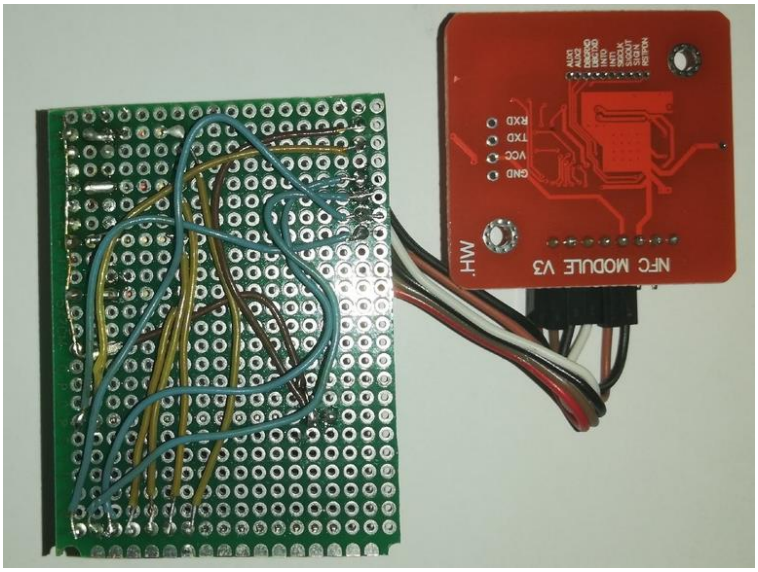

Рисунок 3.9 – Тыльная сторона платы устройства

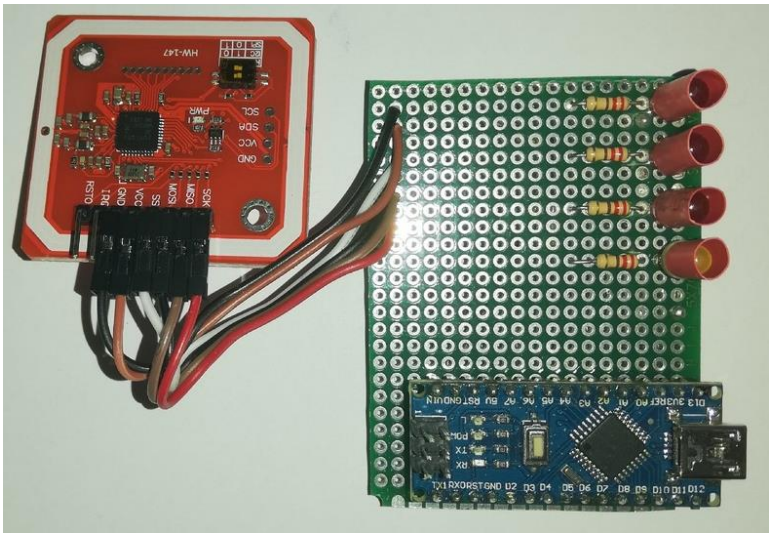

Рисунок 3.10 – Лицевая сторона платы устройства

3.2 ПО для Android

В рамках реализации дипломного проекта следующей значимой задачей является написание приложения на смартфон. Главной функцией данного приложения должна быть возможность организации подключения по NFC смартфона и разработанного устройства.

3.2.1 Написание программы

Прежде чем приступать к написанию программного кода требуется определить некоторые критерии, которым должно соответствовать разрабатываемое приложение, а именно:

1.Приложение должно запускаться на смартфоне с операционной системой Android.

2. В приложении должен быть реализован механизм поддержки NFC модуля устройства.

3. Приложение должно осуществлять проверку на наличие и статус самого модуля NFC, а также включенные технологии NFC Share и Android Beam.

4. Должна присутствовать функция автоматического запуска приложения при взаимодействии смартфона с устройством.

Помимо указанных критериев для понимания работы механизма приложения и облегчения написания программного кода, построю блок-схему алгоритма работы программы (рисунок 3.11).

Рисунок 3.11 отражает работу алгоритма системы проверки наличия и состояния NFC модуля на устройстве. Данная функция соответственно позволяет пользователю, в случае возникновения ошибок при соединении устройства и смартфона запустить самостоятельно приложение и уточнить состояние NFC модуля на устройстве.

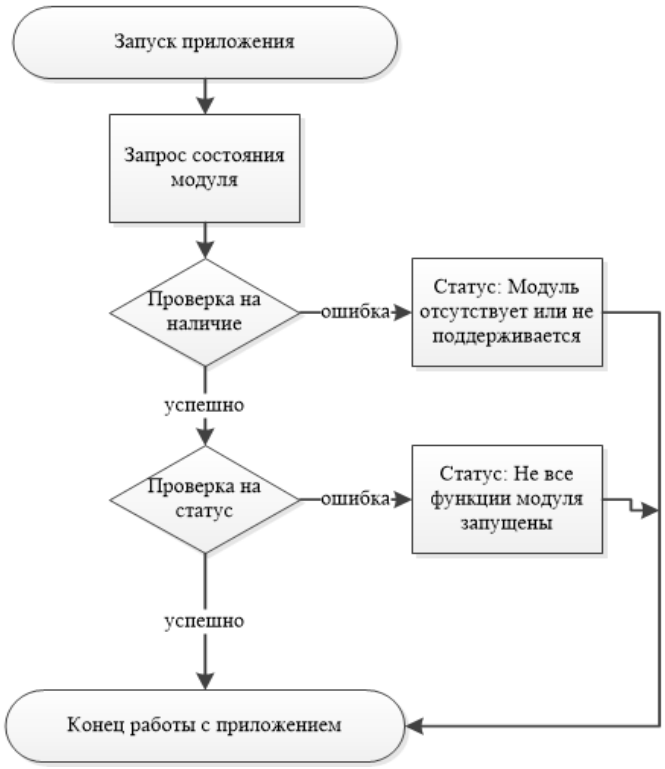

Рисунок 3.11 – Алгоритм работы проверки состояний NFC-модуля

Рисунок 3.12 отражает работу основного алгоритма, который выполняется при осуществлении пользователем процесса взаимодействия смартфона и устройства.

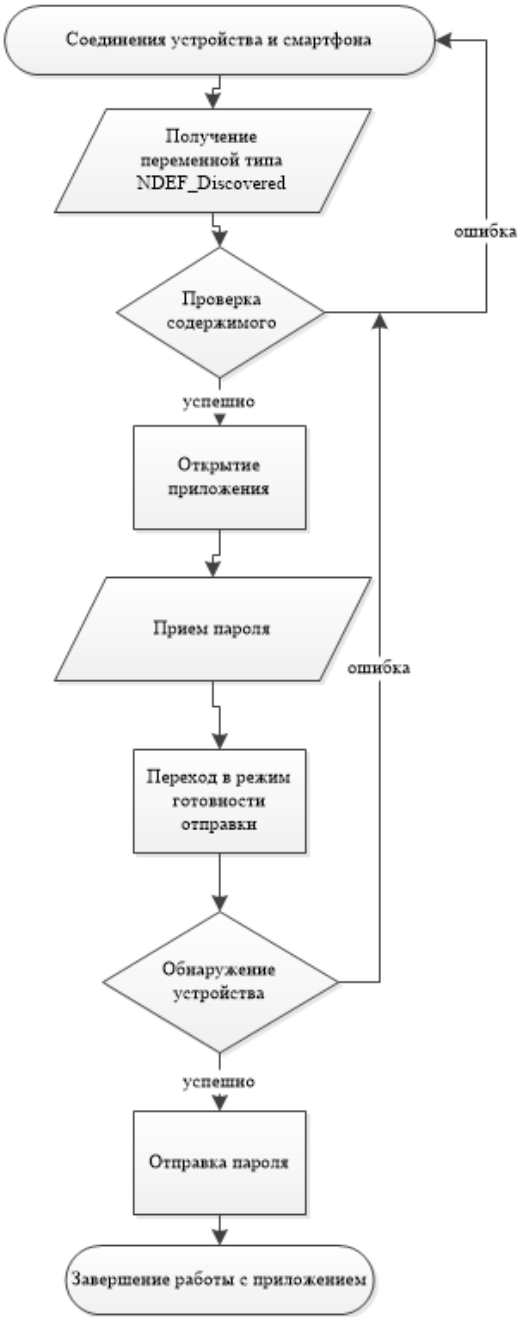

Рисунок 3.12 – Алгоритм работы основной программы при отправке сообщений на устройство

Поскольку я построил алгоритмы работы приложения и точно знаю какие функции оно будет выполнять, можно приступать к написанию программного кода. Рисунки 3.13, 3.14 отражают основные моменты в коде приложения, которым уделялось большое внимание.

```
<activity android:name=".MainActivity"
    android:label="NFCapp">
    <intent-filter>
        //Задание запуска по умолчанию
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <intent-filter>
        //Задание функции открытия, при поступлении сообщения от Устройства на Arduino
        <action android:name="android.nfc.action.NDEF_DISCOVERED" />

        <category android:name="android.intent.category.DEFAULT" />

        <data android:mimeType="text/plain" />
    </intent-filter>
</activity>
```

Рисунок 3.13 – Содержимое Manifest файла

Рисунок 3.14 отображает настройки Manifest файла приложения, которые задаются для правильной реакции приложения при поступлении NDEF-сигнала от устройства к системе диспетчерезации тегов смартфона.

```
//получение статуса NFC на устройстве
mNfcAdapter = NfcAdapter.getDefaultAdapter(this);
//обращаемся к элементу - кнопка
Button chesk = (Button) findViewById(R.id.cont);
if (mNfcAdapter == null) {
    //Прекращаем работу если нет устройства NFC
    Toast.makeText( context: this,  text: "Это устройство не поддерживает NFC!!", Toast.LENGTH_LONG).show();
    finish();
    chesk.setVisibility(View.INVISIBLE);
    return;

}

//Проверяем статус устройства
if (!mNfcAdapter.isEnabled()) {
    //выводим сообщение о статусе и блокируем кнопку ввода
    Toast.makeText( context: this,  text: "NFC не включен", Toast.LENGTH_LONG).show();
    chesk.setVisibility(View.INVISIBLE);
} else {
    chesk.setVisibility(View.VISIBLE);
```

Рисунок 3.14 – Программный код стартовой формы

Программный код, показанный на рисунке 3.14 позволяет приложению запросить статус и наличие NFC-модуля в устройстве для выполнения алгоритма проверки показанного на рисунке 3.11.

```
//Создаем переменную nfc присвеиваем ей параметры адаптера по умолчанию
NfcAdapter nfc = NfcAdapter.getDefaultAdapter(this);
//Задаем тип ивента такой как Callback
nfc.setNdefPushMessageCallback((event) → {
        //Задаем тип данных который будет передан в сообщении
        String message = mTextNFC.getText().toString();
        //Задаем содержимое сообщения
        NdefRecord uriRecord = NdefRecord.createMime( mimeType: "text/plain",message.getBytes());
        //Задаем параметры, которые будет возвращать ивент
        return new NdefMessage(new NdefRecord[] { uriRecord });
}, activity: this, ...activities: this);
```

Рисунок 3.15 – Программный код отправки сообщения от смартфона на устройство

Как видно из рисунка3.15 программный код выполняет преобразование типов данных, для создания NDEF-сообщения. При сопряжении устройства и смартфона экран, отображающий передачу изменяется и благодаря описанному коду, при касании экрана, пользователь осуществяет процесс передачи сообщения.

В приложении Б приведен листинг программы.

При написании данного приложения были задействованы библиотеки:

- import android.content.Intent;
- import android.nfc.NfcAdapter;
- import android.os.Bundle;
- import android.view.View;
- import android.widget.Button;
- import android.widget.EditText;
- import android.widget.TextView;
- import android.widget.Toast.

3.2.2 Скриншоты работы приложения

После успешного завершения этапа установки произвожу ручной запуск приложения для получения статуса работоспособности модуля NFC (рисунок 3.16).

Из рисунка 3.16 видно, что испытания проводятся на программном эмуляторе Nexus 6. Поскольку в Android Studio не существует возможности программной эмуляции работы NFC мы видим следующее сообщение на экране. Перехожу к тестированию программы на реальном устройстве (рисунок 3.17).

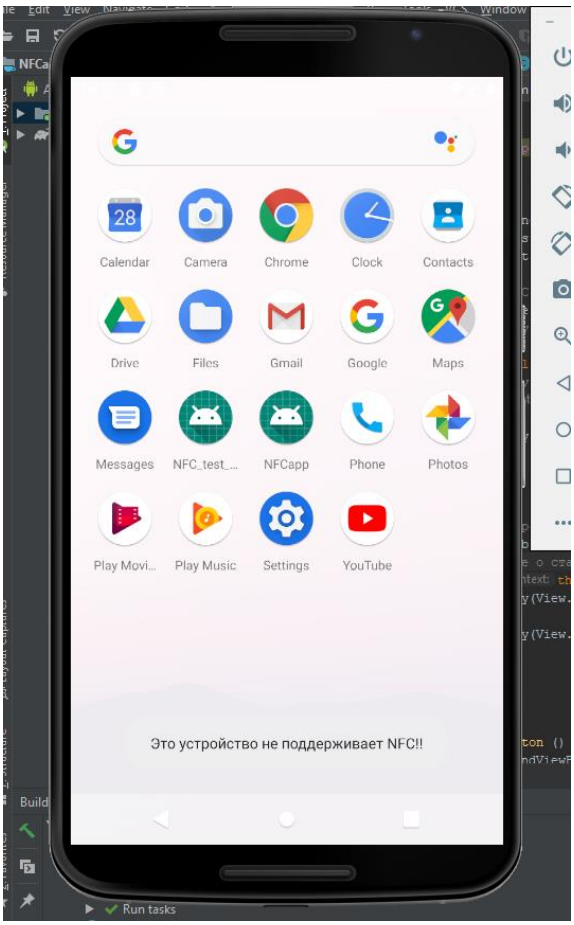

Рисунок 3.16 – Вывод статуса работы приложения

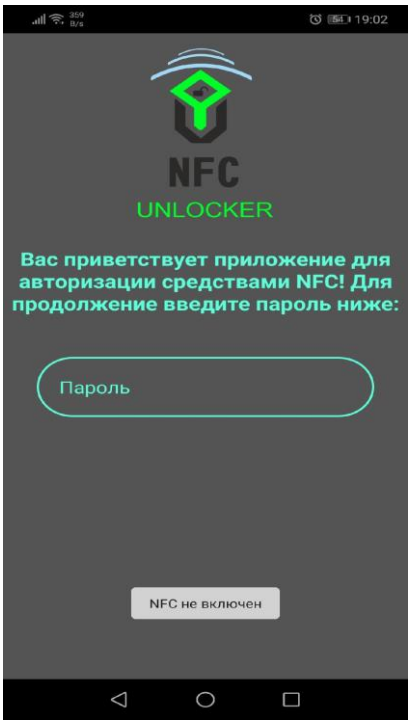

Рисунок 3.17 – Запуск приложения

Из рисунка 3.17 видно, что приложение запустилось некорректно, в нижней части экрана присутствует уведомление о статусе модуля в смартфоне. Для устранения перехожу в настройки и включаю NFC.

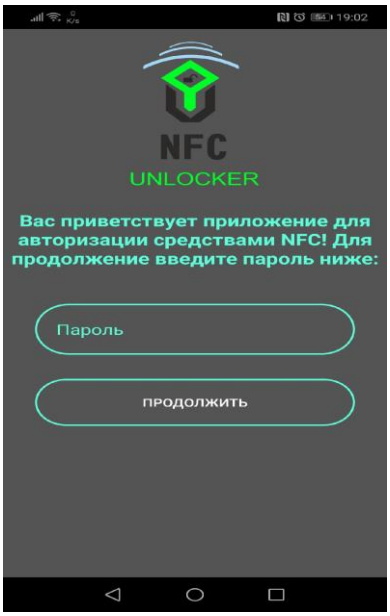

Рисунок 3.18 – Удачный запуск приложения

Как видно на рисунке 3.18 приложение запустилось успешно и присутствуют все элементы начального экрана, предлогающие ввести какое-либо тестовое значение в поле «пароль» и нажать на кнопку «продолжить».

На рисунке 3.19 в нижней части видно уведомление о том, что устройство готово к разблокировке, а это значит, что я могу связать устройство и смартфон при помощи данного приложения.

Рисунок 3.19 – Форма передачи данных

3.3 ПО для Windows

Программное обеспечение, разрабатываемое для операционной системы Windows предназначено для автоматизации процессов настройки файлов конфигурации, сохранении данных в файлах, в корректной кодировке (UTF-8), автоматического запуска файлов вносящих изменения в реестр, а также упрощении процесса контроля релевантности изменений, вносимых в файлы конфигурации механизма аутентификации.

Смысл работы механизма проще можно объяснить на рисунке 3.20.

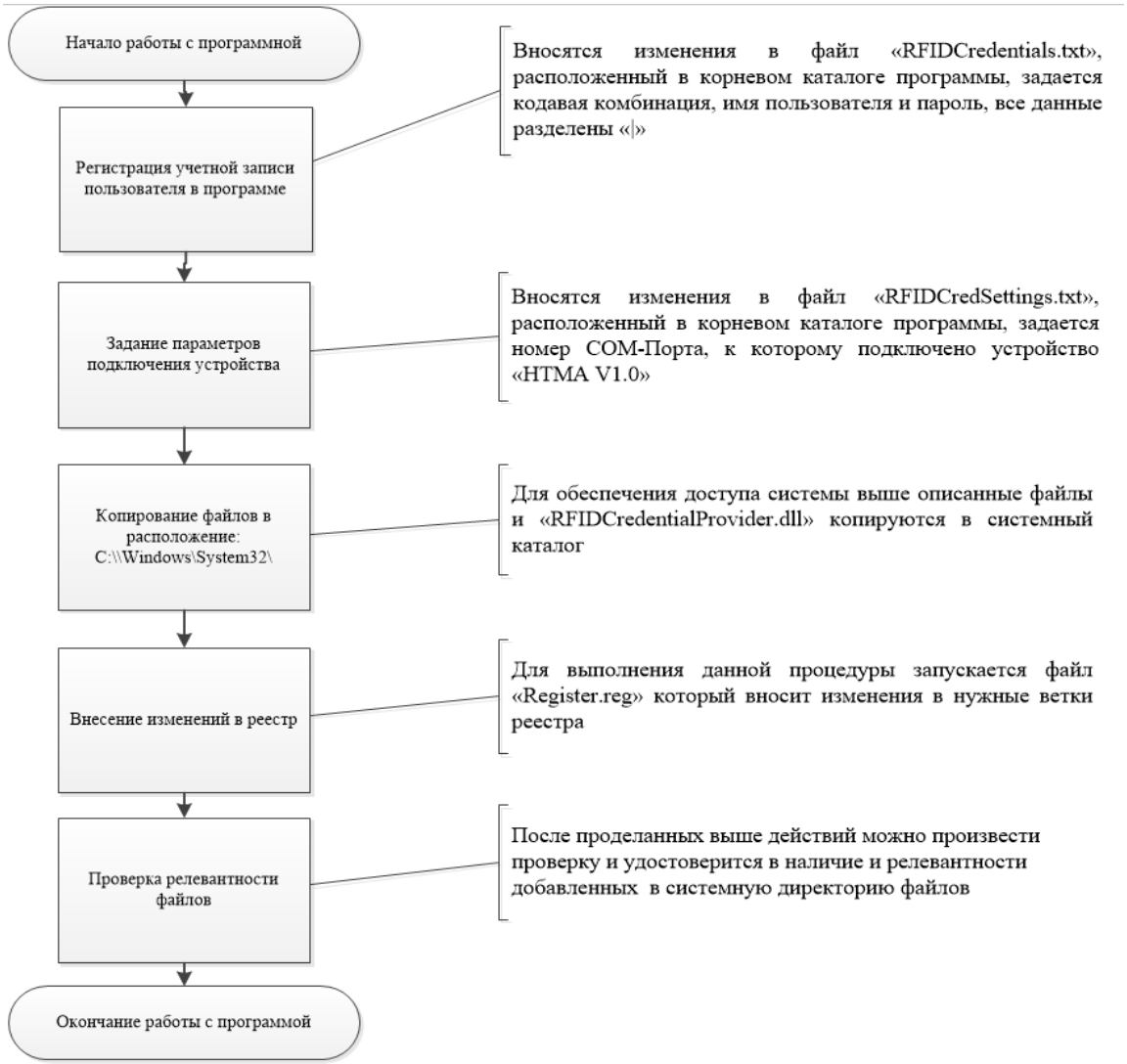

Рисунок 3.20 – Функциональная схема работы программы

3.3.1 Написание программы

Для облегчения настройки всех параметров Windows, которые позволят обеспечить аутентификацию пользователя, используя смартфон и устройство NFC, разрабатывается программа. В базовый набор ее функций должны входить:

1. Создание, хранение и редактирование базы пользователей с их учетными данными;

2. Внесение и удаление изменений в разделы реестра, отвечающие за механизм авторизации пользователя;

3. Создание и хранение файла конфигурации соединения между компьютером и устройством «НТМА v1.0».

Прежде всего составлю алгоритм работы базовых функций программы (рисунки 3.21, 3.22).

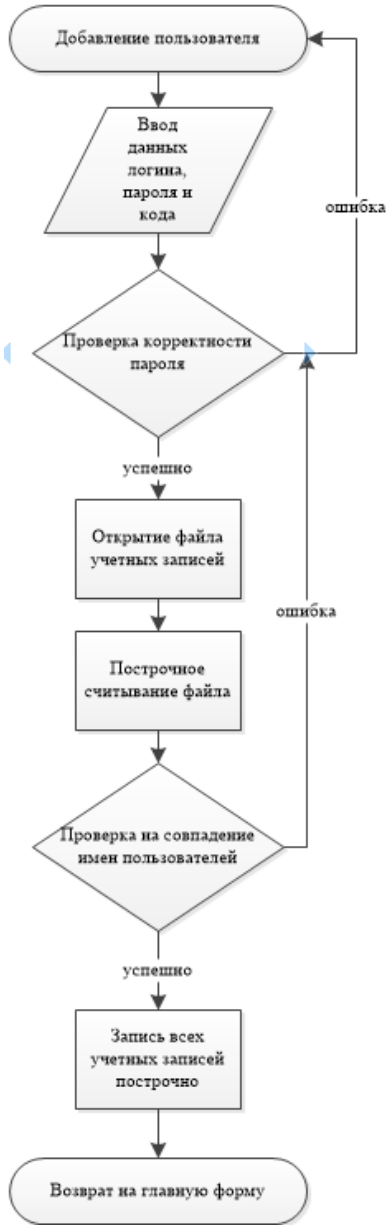

Рисунок 3.21 – Блок-схема добавления нового пользователя

На рисунке 3.21 показан алгоритм добавления нового пользователя. Весь смысл данной операции заключается в редактировании файла с именем «RFIDCredentials.txt» (рисунок 3.22).

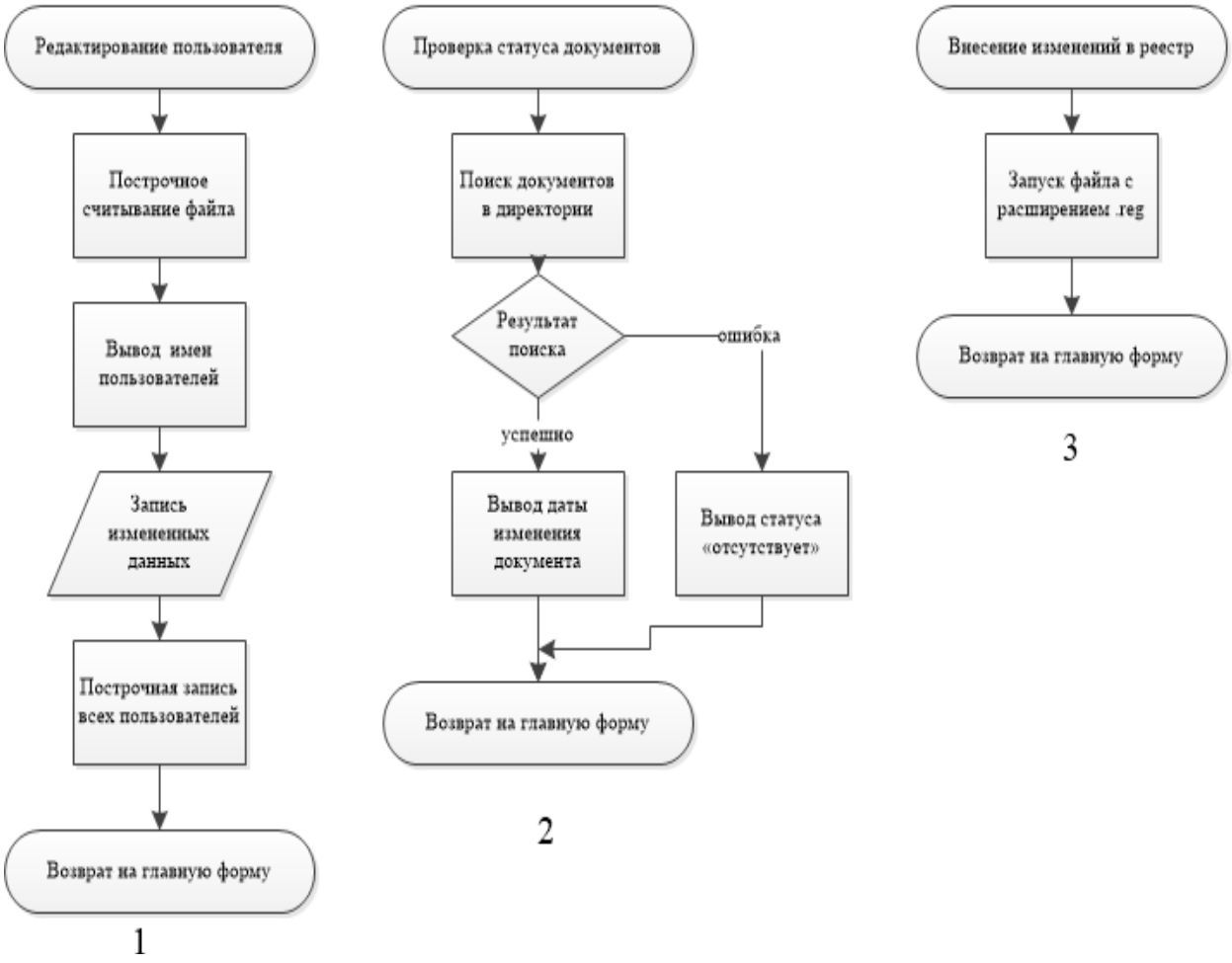

Рисунок 3.22 – Блок-схемы алгоритмов функций программы

На рисунке 3.22 показано 3 блок-схемы, каждая из которых отображает весь смысл работы одной функции программы. Разберу подробнее. Первая из трех отображает алгоритм, который выполняется при запуске функции редактирования учетной записи пользователя, которая уже была зарегестрирована в программе. Изменения вносятся в файл «RFIDCredentials.txt». Вторая показывает алгоритм работы механизма проверки статуса документа, поскольку файлы, с которыми работает программа должны быть размещены в директории «System32». Данная функция позволяет узнать, когда последний раз вносились изменения в файлы конфигурации. Третья схема показывает как происходит работа механизма внесения изменений в реестр, для обеспечения работы механизма аутентификации средствами NFC.

Используя вышеописанные алгоритмы, описывающие механизм работы приложения, произвожу написание программного кода. Ниже будут отображены ключевые моменты кода программы (рисунки 3.23-3.27).

```
private void button1_Click(object sender, EventArgs e)
{   //Объявляем перененные данных пользователя
    String Name = textBox1.Text;
    String Pass = textBox2.Text;
    String PassConf = textBox3.Text;
    String Fast = textBox4.Text;
    //Обнуляем чекер
    int indexOfSubstring = -1;
    //Находим путь к файлу с которым будем работать
    var exePath = AppDomain.CurrentDomain.BaseDirectory;//path to exe file
    var path = Path.Combine(exePath, "RFIDCredentials.txt");
    //Проверка на совпадение паролей
    if (Pass != PassConf)
    {   MessageBox.Show("Пароли не совпадают!!");
        textBox2.Clear();
        textBox3.Clear();
    }
    else
    {   Encoding code = new UTF8Encoding(false); //Выставляем обязательную кодировку в UTF8 без дополнительных параметров и конфигураций
        string f3 = File.ReadAllText(path, code); //Задаем путь к файлу, в котором хранятся данные

        indexOfSubstring = f3.IndexOf(Name);// проверка на схожие логины пользователей
        if (indexOfSubstring >= 0)
        {                   MessageBox.Show("Такой пользователь уже существует");               }
        else
        {       //Процесс записи данных используя поток
            using (Stream stream = File.OpenWrite(path))
            using (var writer = new StreamWriter(stream, new UTF8Encoding(false)))
            {
                if (f3.Length!=0)
                {                           writer.Write(f3.ToString());                    }

                writer.WriteLine(Fast + '|' + Name + '|' + Pass);
                MessageBox.Show("Пользователь успешно добавлен!"); //вывод сообщения об успешном добавлении пользователя
            }               }           }
    //Возвращение к исходной форме
    Close();
    Form1 newMain = new Form1();
    newMain.Show();
```

Рисунок 3.23 – Код добавления добавления пользователя

На рисунке 3.23 изображен листинг кода, который позволяет добавить пользователя в текстовый файл «RFIDCredentials.txt», который в последствии будет использован для авторизации пользователя.

```
//Запись данных из текстбоксов в списки
Fas.Insert(ind, textBox6.Text.ToString());
Pas.Insert(ind, textBox5.Text.ToString());
//Объявление пути к файлу
var exePath = AppDomain.CurrentDomain.BaseDirectory;
var path = Path.Combine(exePath, "RFIDCredentials.txt");
//Задание исходной кодировки файла и считывание в строковый массив
Encoding code = Encoding.UTF8;
string[] f3 = File.ReadAllLines(path, code);
//замена данных массива с обращением по индексу
f3[ind] = Fas[ind].ToString() + '|' + Nam[ind].ToString() + '|' + Pas[ind].ToString();
//процесс записи изменений в файл построчно
using (Stream stream = File.OpenWrite(path))
using (var writer = new StreamWriter(stream, new UTF8Encoding(false)))
{
    for (int i = 0; i < f3.Length; i++)
    {
        writer.WriteLine(f3[i]);
    }
    writer.Close();
}

MessageBox.Show("Пользователь изменен!");

Close();
Form1 newMain = new Form1();
newMain.Show();
```

Рисунок 3.24 – Код изменения пользовательских данных

На рисунке 3.24 изображен листинг кода, который позволяет произвести построчное считывание данных текстового файла в строковый массив, после чего производится замена конкретной строки полученного массива с обращением по индексу. По завершении всех манипуляций с массивом, выполняется процесс записи с затиранием предыдущих строк.

```
var exePath = AppDomain.CurrentDomain.BaseDirectory;//Задание пути к исполняемому файлу программы
var path = Path.Combine(exePath, "Register.reg");//Задание пути к файлу вносящему изменения в реестр
var proc = new Process();//Создаем переменную процесса
var pi = new ProcessStartInfo();//Создаем переменную вывода статуса о процессе
pi.UseShellExecute = true;//Отключаем вывод консоли
pi.FileName = path;//Задаем пусть к файлу, редактирующему реестр
proc.StartInfo = pi;//Передаем параметры запуска
proc.Start();//Запускаем файл вносящий изменения в реестр
```

Рисунок 3.25 – Код запуска файла с расширением «reg»

Код, показанный на рисунке 3.25 позволяет произвести запуск файла с расширением «reg». Однако, для полноценной и корректной работы этой части кода требуется сгенерировать файл manifest для разрабатываемой программый и указать тип запуска, иными словами говоря, поставить требование запуска от Администратора. На рисунке 36 показан сегмент файла manifest, который нужно изменить.

```
<?xml version="1.0" encoding="utf-8"?>
<assembly manifestVersion="1.0" xmlns="urn:schemas-microsoft-com:asm.v1">
  <assemblyIdentity version="1.0.0.0" name="MyApplication.app"/>
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v2">
    <security>
      <requestedPrivileges xmlns="urn:schemas-microsoft-com:asm.v3">
        <requestedExecutionLevel  level="requireAdministrator" uiAccess="false" />
      </requestedPrivileges>
    </security>
  </trustInfo>
```

Рисунок 3.26 – Содержимое manifest-файла программы

```
//копирование файла конфигурации считывания
var exePath = AppDomain.CurrentDomain.BaseDirectory;
var path = Path.Combine(exePath, "RFIDCredSettings.txt");
var conPath = (@"C:\Windows\System32\RFIDCredSettings.txt");
File.Copy(path,conPath, true);

//копирование файла пользователей
var exePath2 = AppDomain.CurrentDomain.BaseDirectory;
var path2 = Path.Combine(exePath2, "RFIDCredentials.txt");
var conPath2 = (@"C:\Windows\System32\RFIDCredentials.txt");
File.Copy(path2, conPath2, true);

//копирование dll файла
var exePath3 = AppDomain.CurrentDomain.BaseDirectory;
var path3 = Path.Combine(exePath3, "RFIDCredentialProvider.dll");
var conPath3 = (@"C:\Windows\System32\RFIDCredentialProvider.dll");
File.Copy(path3, conPath3, true);
```

Рисунок 3.27 – Код, выполняющий копирование необходимых файлов

Сегмент кода, изображенный на рисунке 3.27 позволяет произвести копирование сконфигурированных файлов программы в системную директорию «C:\Windows\System32\». Однако для корректной работы механизма копирования требуется точно задать разрядность системы, на которой будет работать приложение. Задать данные настройки возможно нажав правую кнопку мыши по проекту, и выбрать пункт «Свойства». В открывшемся окне нужно задать параметры, как показано на рисунке 3.28.

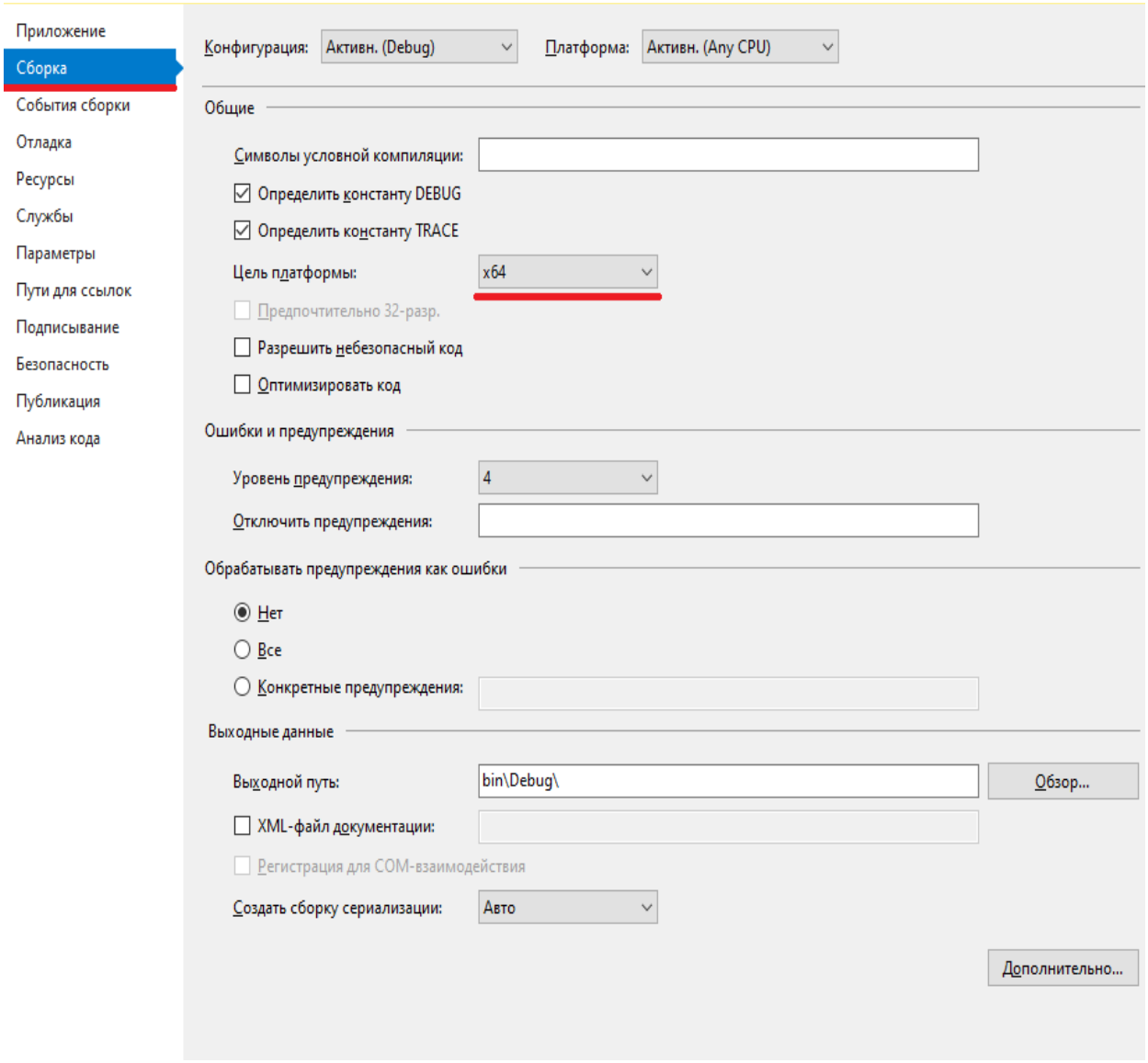

Рисунок 3.28 – Настройка разрядности приложения

В приложении В приведен листинг программы.

При написании данного приложения были использованы библиотеки:

- using System;
- using System.Collections.Generic;
- using System.ComponentModel;
- using System.Data;
- using System.Drawing;
- using System.Linq;
- using System.Text;
- using System.Threading.Tasks;
- using System.Windows.Forms;
- using System.IO.Ports;
- using System.IO;
- using System.Diagnostics.

3.3.2 Скриншоты работы

Для корректной работы к оборудованию при работе в программе NFCUlocker выдвигаются следующие системные требования:

- операционная система: Windows 7x64 и выше;
- наличие свободного пространства на диске: 15 Мб;
- объем свободной оперативной памяти не менее: 30 Мб;
- наличие прав администратора у пользователя;
- предустановка .NetFramework 4.6.1 или более нового.

По завершении процесса установки программы перед пользователем откроется стартовое окно программы, показанное на рисунке 3.29.

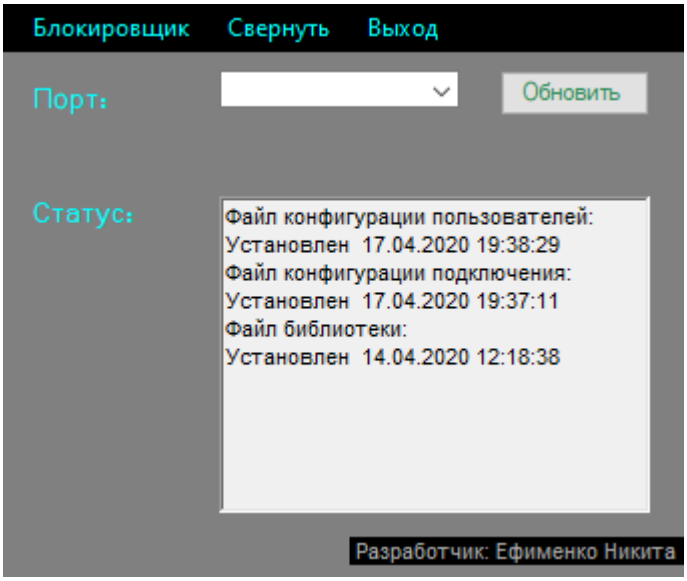

Рисунок 3.29 – Главное окно программы NFCUnlocker

Вся работа с программой завязана с данным окном, за исключением режимов добавления и редактирования пользователей. Как видно из рисунка 3.29, в распоряжении имеется верхняя панель с размещенными кнопками «Блокировщик», «Свернуть», «Выход», также в основной части окна находятся выподающий список, кнопка «Обновить» и большой белый прямоугольник, выводящий статус файлов конфигурации. Разберу кнопки подробнее. Кнопка «Свернуть» из верхней панели позволяет убрать программу в трей, она показана на рисунке 3.30.

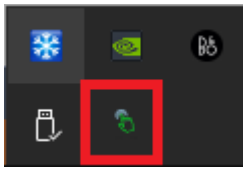

Рисунок 3.30 – Иконка программы в трее

При одиночном нажатии левой кнопки мыши по иконке программа возвращается в исходное состояние. Кнопка «Выход» соответственно своему названию отвечает за выход из программы.

Нажимая на кнопку «Блокировщик», далее открывается выпадающее меню, состоящее из трех пунктов: «Работа с учетными данными», «Реестр», «Обновление параметров». Пункт «Работа с учетными данными» предназначен для организации процессов добавления, изменения или удаления пользователя из базы данных программы. На рисунках 3.31-3.33 будут показаны окна, отвечающие за перечисленные функции.

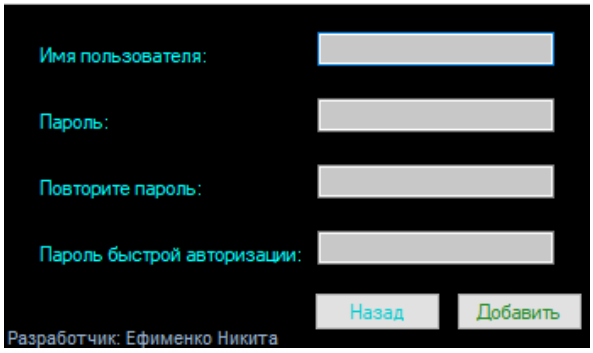

Рисунок 3.31 – Окно добавления нового пользователя

После заполнения полей окна программы в корневой директории программы происходит создание файла «RFIDCredentials.txt». Структура этого файла отображена ниже.

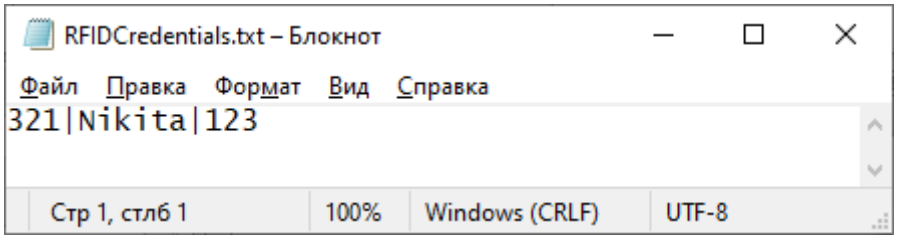

Рисунок 3.32 – Содержимое файла «RFIDCredentials.txt»

Как видно из рисунка части данных разделены символом «|», если говорить точно о данной записи, то в первой ее части расположен короткий код доступа, используемый при аутентификации средствами устройства «НТМА v1.0». Далее идет имя пользователя и пароль учетной записи этого пользователя.

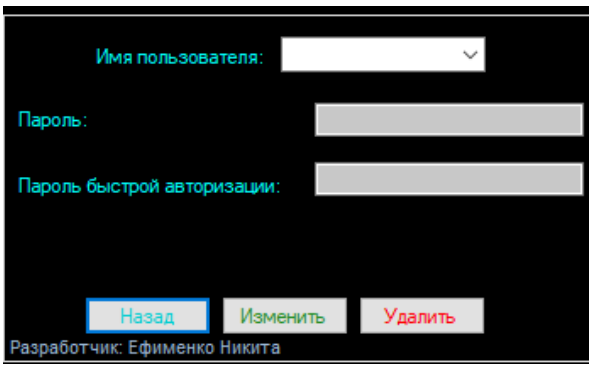

Рисунок 3.33 – Окно редактирования / изменения пользователя

Рисунок 3.33 отобраает внешний вид окна редактрора учетных данных пользователя. Проще говоря, данное окно позволяет вносить изменения в файл «RFIDCredentials.txt»

Пункт меню «Реестр» позволяет выбрать один из двух вариантов, что более детально показано ниже (рисунки 3.34-3.35).

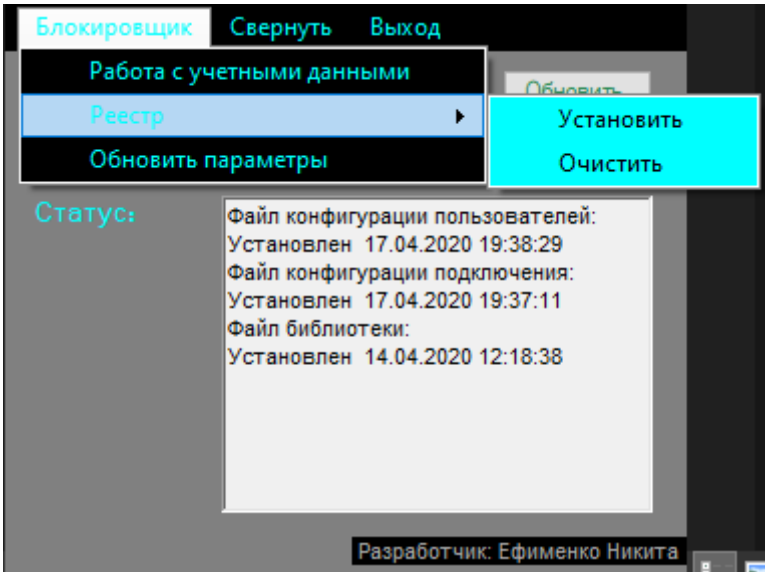

Рисунок 3.34 – Работа с реестром

Пункты меню «Установить» и «Очистить» относятся к работе с реестром в программе. Соотвественно своим названиям они вносят или убирают внесенные изменения в дереве реестра. Данные изменения вносятся для корректировки работы службы WCP, определении механизма, который будет отвечать за аутентификацию пользователя в системе.

Последний пункт списка – это «Обновить параметры». Он позволяет переносить файлы, требуемые для работы механизма аутентификации в системную директорию по адресу «C:\Windows\System32\».

С разбором функций верхнего меню основного окна программы мы закончили, перейдем к двум оставшимся объектам. Кнопка обновить предназначена для обновления данных поля статус и данных из

выподающего меню порт. Соответственно выпадающее меню «порт», требуется для задания COM-порта, в качестве которого определилось устройство. Рисунок 3.35 покажет содержимое выпадающего списка при подключенном устройстве.

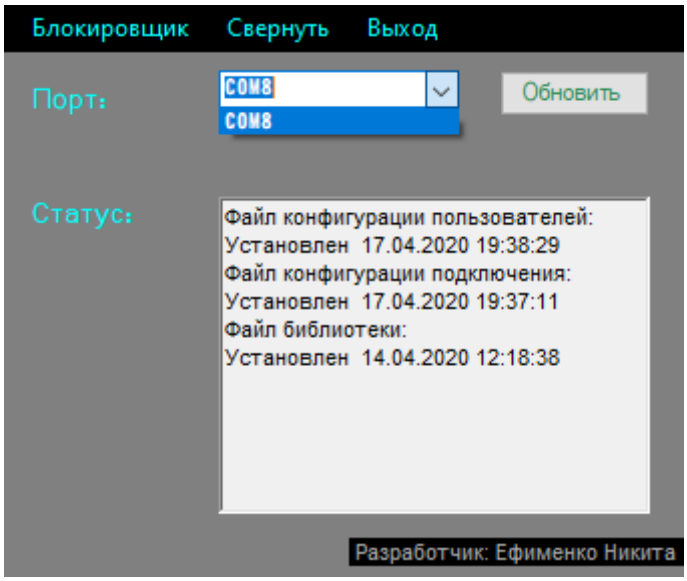

Рисунок 3.35 – Главное окно программы NFCUnlocker

3.4 Тестирование проекта

После завершения разработки всех составляющих проекта перехожу к тестированию. Для наглядности понимания механизма работы всего проекта напишу схему (рисунок 3.36).

Рисунок 3.36 – Схема поэтапной работы проекта

Первым делом, осуществив настройку компьютера под управлением Windows средствами утилиты «NFCUnlocker», произвожу его блокировку или выключение.

Далее для получения доступа потребуется смартфон с установленным приложением «NFCapp». Подношу смартфон как можно ближе к устройству «НТМА v1.0» и жду звукового уведомления отрабатывании системы диспетчеризации тегов, после этого должно произойти автоматическое открытие приложения на смартфоне и на устройстве «НТМА v1.0» должен замигать зеленый светодиод.

После завершения индикации зеленого светодиода, должен зажечься красный, обозначенный как «Режим приема». Это означает что идентификация устройства проидена успешно и можно ввести пароль на смартфоне.

По окончинию ввода нажимаю кнопку «Продолжить». На экране смартфона появляется картинка с уведомлением, что устройство готово к разблокировке.

Далее как и в первый раз, подношу смартфон максимально близко и касаюсь экрана смартфона. На этом работа со смартфоном завершена, наблюдаю за индикаторами на устройстве, если зеленый снова начал моргать, значит процесс авторизации успешно пройден и доступ к системе получен. Можно переходить к работе с компьютером.

Однако не всегда все может протекать идеально и в некоторых случаях могут возникать ошибки в работе, для этого и предназначен красный индикатор, обозначенный на устройстве «НТМА v1.0» как «Ошибка». Ниже на рисунке 3.37 дана схема аналогичная верхней, но отражающая ошибки, возникающие при работе.

Рисунок 3.37 – Схема поэтопной работы с выделением ошибок

Из рисунка 3.37 видно, что могут возникать 2 типа ошибок, первый тип возникает при изначальном распознавании устройства, как правило он может возникать из-за несовметимых чипов NFC-контроллеров, либо из-за большого расстояния между устройством и смартфоном. Для решения данной ошибки как правило требуется просто немного подождать и снова поднести смартфон к устройству.

Другая ошибка может возникать при синхронизации механизма передачи сообщения со смартфона на устройство, как и в первой ошибке, она может возникать из-за расстояния между устройством и смартфоном. Также она может возникать из-за неизвестной аппаратной ошибки, возможно связанной со скоростью обработки информации платой Arduino NANO, поскольку данная ошибка не возникает при использовании микроконтроллера Arduino Mega, которая ощутимо работает быстрее. Особенна заметна скорость работы при выполнении перезагрузки устройства в момент индикации загрузки светодиодами, Mega производит данный процесс гораздо быстрее.

Помимо выявленных вышеописанных ошибок, которые фиксирует само устройство Arduino, присутствует ошибка распознавания устройства, если внсти изменения в учетную запись пользователя и при этом не произвести операцию очистки и установки параметров в реестр, то механизм WCP отказывается работать с устройством. Это означает, что авторизация на устройстве может пройти успешно, однако в дальнейшем доступа к системе не будет. На рисунках ниже показана данная ошибка.

Рисунок 3.38 – Устройство «НТМА» с усаешным результатом авторизации

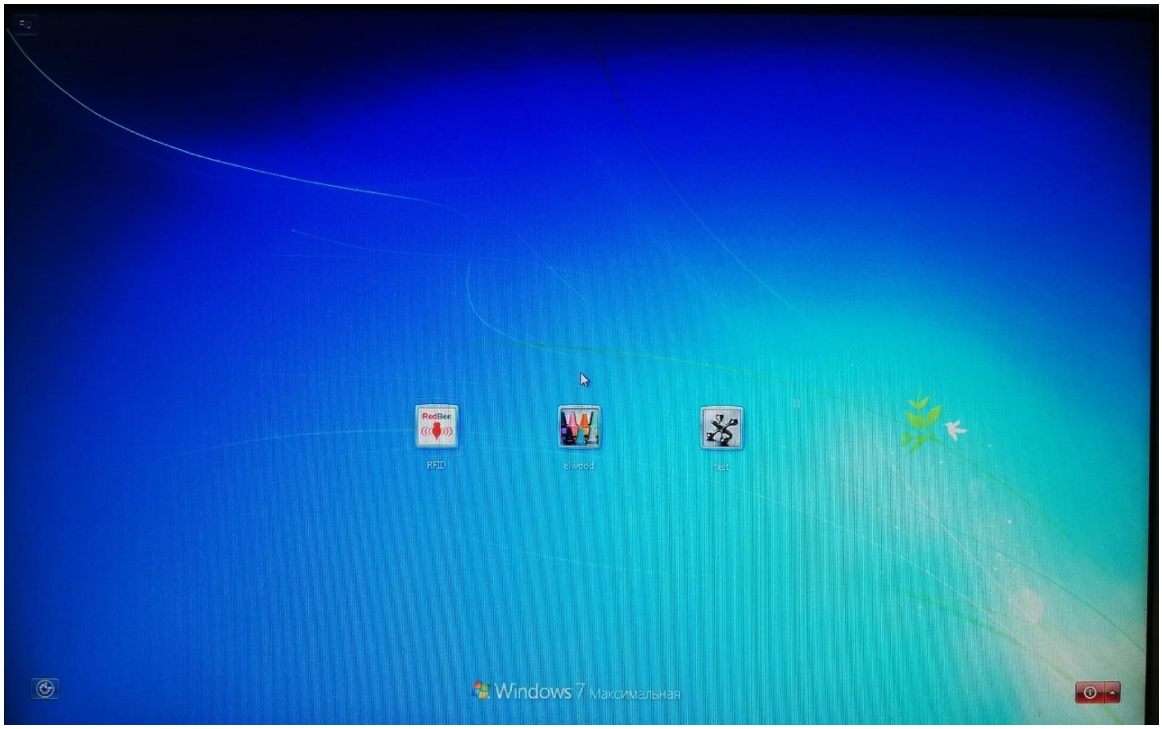

Рисунок 3.39 – Изображение на авторизации на мониторе компьютера

Вывод

В данной главе описывается детальная разработка всех составляющих проекта, а именно: устройства, описываются аппаратные компоненты, применяемые в реализации программно-аппартаной части устройства. Помимо этого, указываются тонкости и интересные моменты, ошибки, которые возникали в процессе проведенных работ, типизируются типы ошибок и даны были рекомендации по их устранению. Описаны были физические и программные характеристики устройства и программ на разные платформы. Рассматривался функционал и возможности программно-аппаратной части.

Завершающей частью главы является тестирование полученного комплекса. Мной был детально расписан алгоритм взаимодействия пользователя с проектом, для осуществления работы механизма аутентификации базирующемся на технологии NFC. Однако как и в любом проекте, уделяется внимание возникающим в процессе тестирования ошибкам, которые так или иначе связаны с физическими ограничениями используемых ресурсов.

По итогу можно сказать, что полученный комплекс позволяет при правильной настройке производить процесс аутентификации с использованием технологии NFC, то есть цель, поставленная в начале дипломного проектирования успешно выполнена: я создал устройство, написал на него прошивку, создал ПО для Android, ПО для Windows; протестировал работу всех составляющих, внес необходимые корректировки и собрал финальный вариант, тем самым наглядно произвел (используя процесс пайки и сборки вживую) процесс аутентификации с использованием технологии NFC.

4 Расчет рисков

Расчет рисков безопасности, возникающих на производстве, немаловажный аспект в огранизации процесса построения качественного периметра безопасности. Существует множество разнообразных вариантов организации подхода к процессу расчета рисков.

В качестве среды, для которой будут производится расчеты рисков используется организация, в которой имеется головной офис. В штате головного офиса работают несколько человек. Все они работают за личными машинами под управлением ОС Windows. Предположим, что глава компании решает произвести внедрение в головной офис разработанный программно-аппаратный комплекс. Ставится задача, произвести расчеты рисков всех активов, которые так или иначе связаны с работой комплекса. Расчеты требуется произвести до и после внедрения в рабочую среду.

4.1 Описание методики расчета

Для выполнения данной части проекта используется методика расчета по двум параметрам. В качестве параметров над которыми будут производится расчеты, используются:

-Коэфициенты показателей «Конфиденциальности», «Целостности», «Доступности»;

-Коэфициент уровня угрозы;

-Коэфифицент показателя степени уязвимости.

Согласно стандарту BS 77990 уровень риска вычисляется с учетом трех показателей – ценности ресурса, уровня угрозы и степени уязвимости. Формульный вид представлен ниже:

$$R = S * L(t) * L(v) \tag{4.1}$$

Где R - значение риска;

S - ценность актива/ресурса;

L(t) - уровень угрозы;

L(v) - уровень/степень уязвимости.

После расчета показателей уровня риска, производится введение мер, препятствующих реализации риска, в данном случае этими мерами должно служить введение программно-аппаратного комплекса. Перерасчет остаточного показателя риска после введения мер осуществляется по формуле ниже:

$$R_{\text{ост}} = S * L(t)_{\text{ост}} * L(v)_{\text{ост}} \tag{4.2}$$

Где $R_{ост}$ - остаточный риск;
S - ценность актива/ресурса;
$L(t)_{ост}$ - уровень угрозы после введения мер;
$L(v)_{ост}$ - уровень/степень уязвимости после введения мер.

По итогам расчетов должно сформироваться мнение, у специалиста, который производил расчеты, о эфективности вводимых мер защиты.

4.2 Проведение расчетов

Первым делом произведем описание присутствующих в организации основных активов, на которые может распространятся действие программно-аппаретного комплекса диплосного проекта. Выполним построение диаграмм в программе Coras.

На рисунке 4.1 показаны активы, и потоки данных, которые протекают между этими активами. Относительно проекта выделено 2 основных актива, это «Вычислительная машина» и «Операционная система». Потоки между данными активами протекают через косвенный актив «Работник организации». Говоря о простых примеров потоков данных можно отметить работу пользователя над какими-либо электронными докусентами, которые представляют некоторую ценность для организации. После описания перечня активов, можно приступать к рассмотрения списка пар угроза + уязвимость, которые могут возникать для выбранных активов.

Рисунок 4.2 описывает пары угроз и уязвимостей, а также к каким активам, по какому сценарию и кем реализуются данные пары. Графическое представления данной диаграммы позволяет более наглядно понять, насколько важен процесс анализа угоз и уязвимостей в организации.

После описания пар угроз и уязвимостей, можно перейти к построению диаграммы рисков, возникающих в организации относительно выбранных активов. Рисунок 4.3 как раз и отображает данную диаграмму, описывая основные риски, активы к которым они относятся, а также источники возникновения данных рисков.

Процесс работы с диаграммной рисков на этом не заканчивается, поскольку относительно выбранных методик расчетов, требуется произвести расчеты. Как видно из рисунка 4.4 каждому риску выставляется определенный числовой показатель, сопровождаемый параметром приемлимости риска для организации.

Когда произведен первичный расчет рисков в организации, можно переходить к процессу выбора мер по оработке риска. Рисунок 4.5 отображает диаграмму 4.2 с наложенными на нее мерами по обработке рисков. Как видно из рисунка, количество мер немного ментьше чем количество пар угроз и уязвимостей, это обусловленно возможностью закрытия одной мерой нескольких пар.Рисунок 4.6 аналогичен 4.4, однако как и в случае с рисунком 4.5 на него наложены меры, которые помогают уменишить показатель получаемый при повторном расчете.

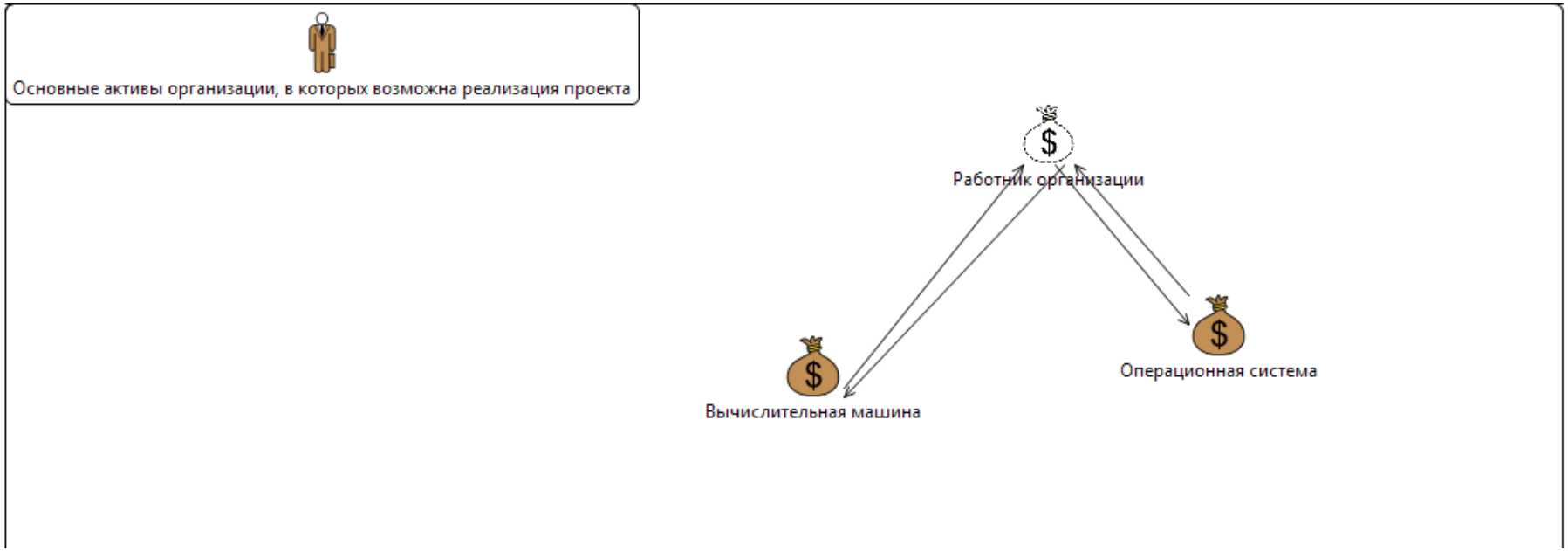

Рисунок 4.1 – Перечень активов

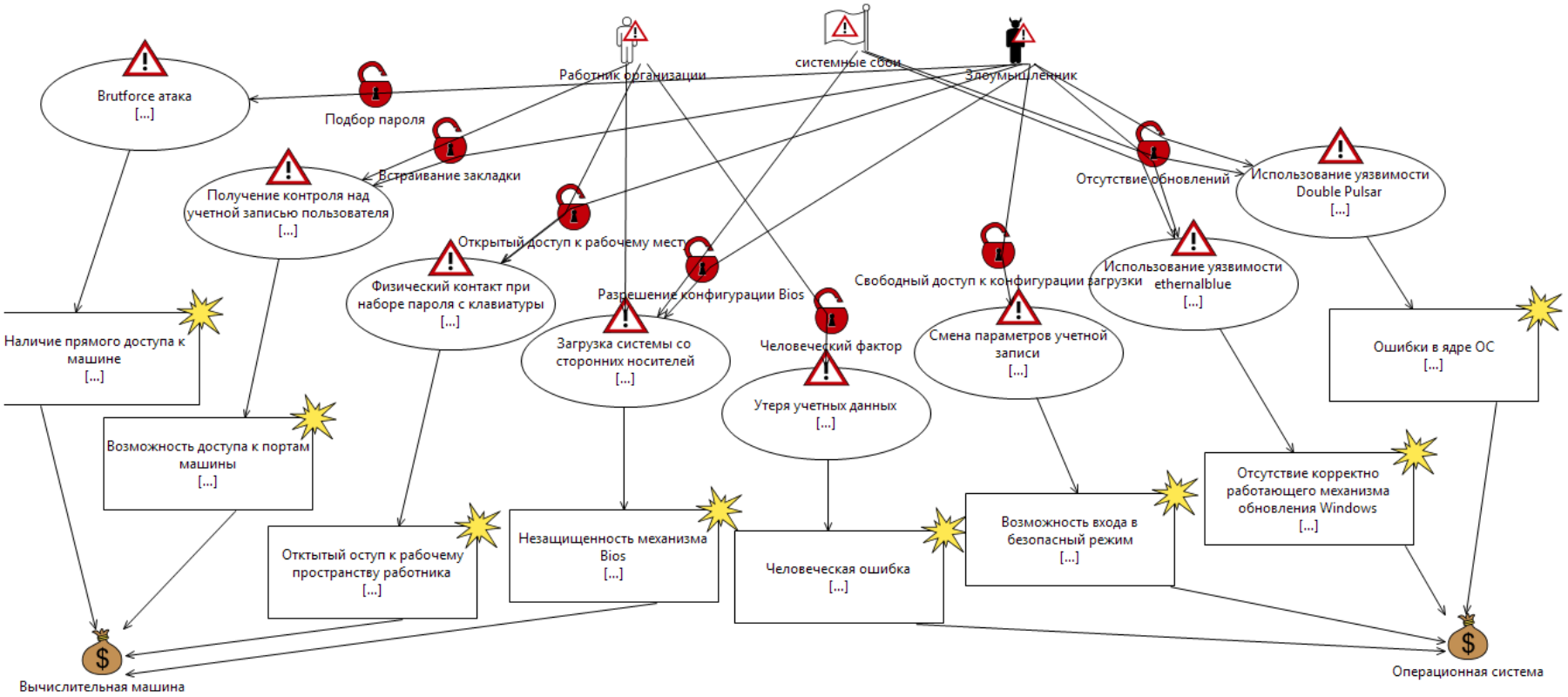

Рисунок 4.2 – Описание пары угроза + уязвимость

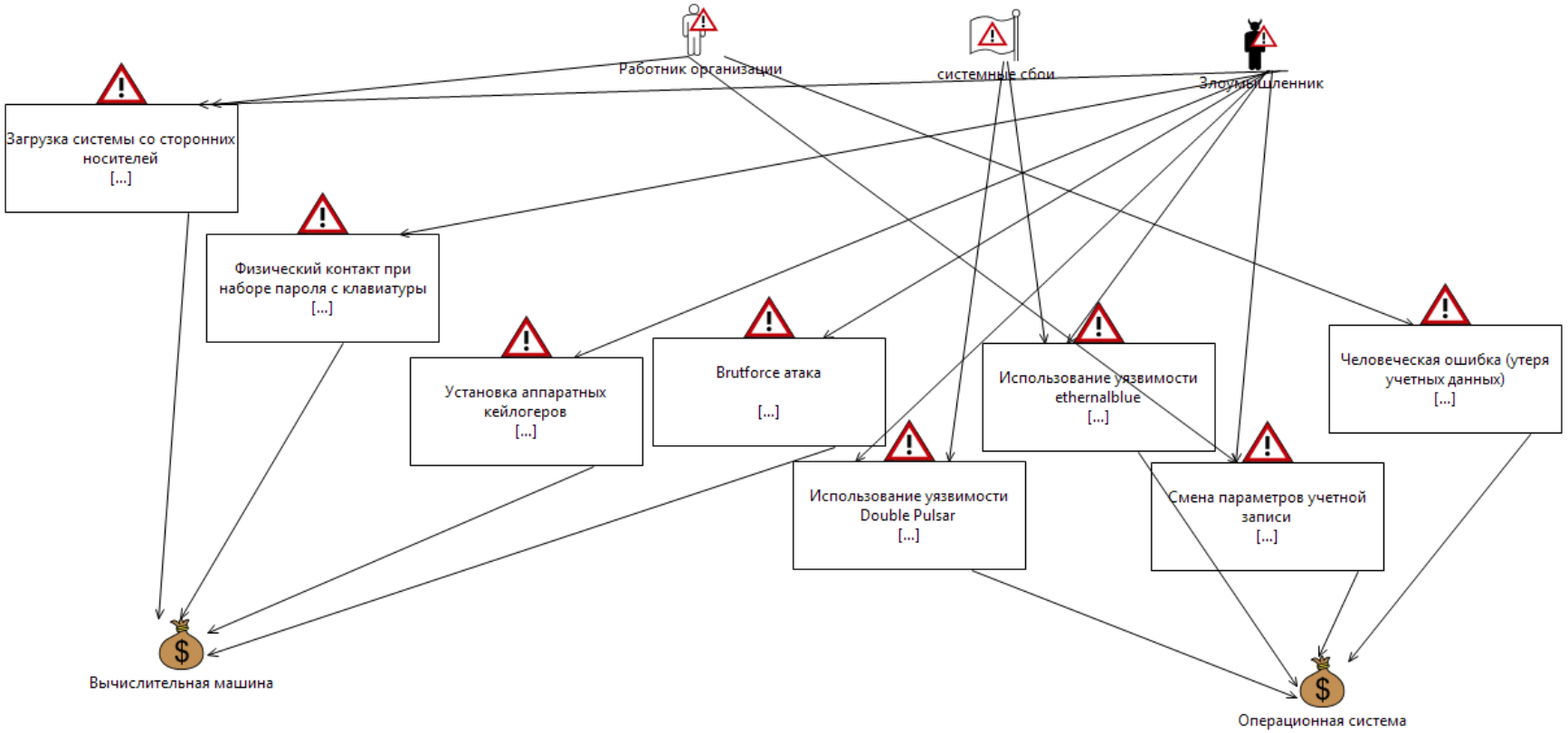

Рисунок 4.3 – Диаграмма рисков

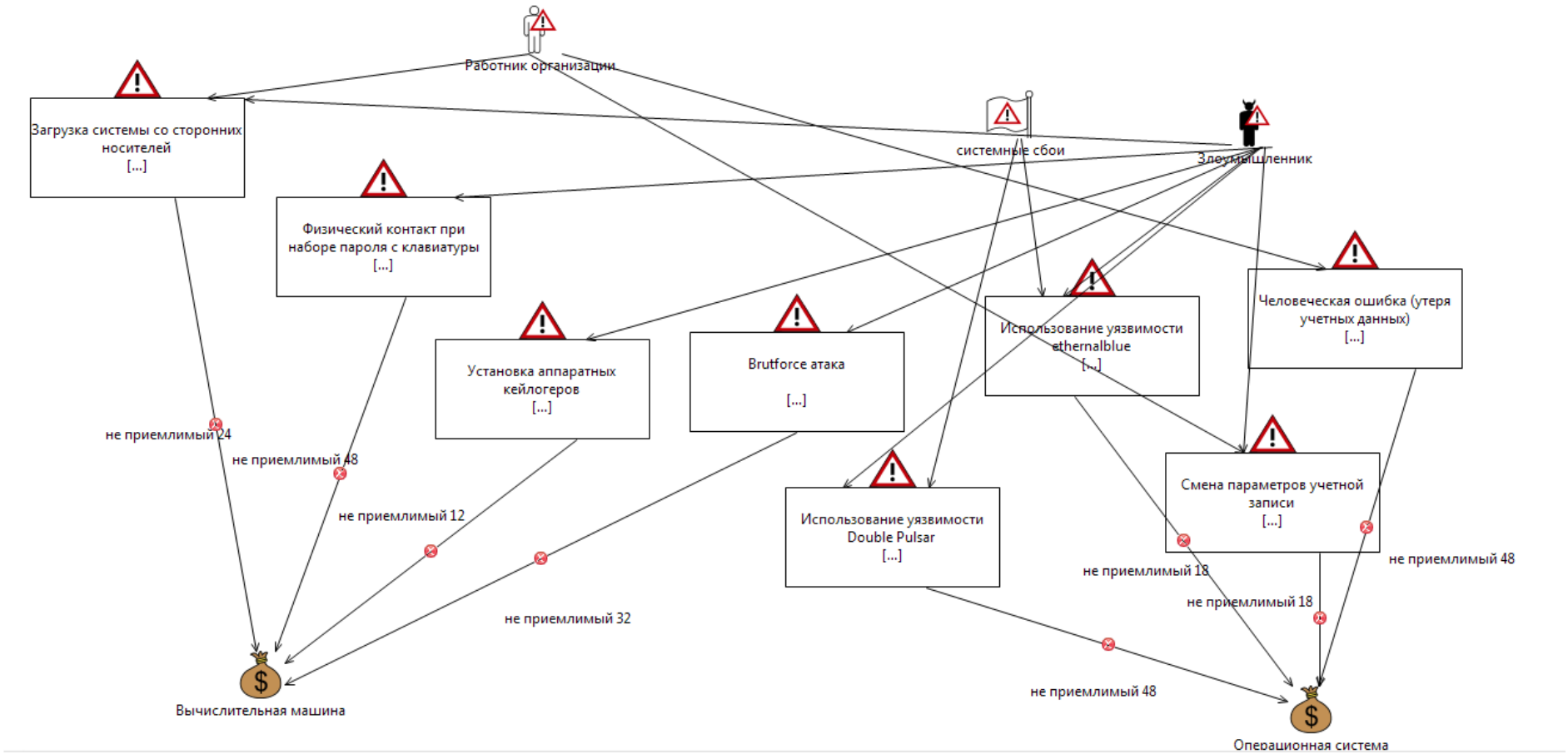

Рисунок 4.4 – Диаграмма рисков с оценкой

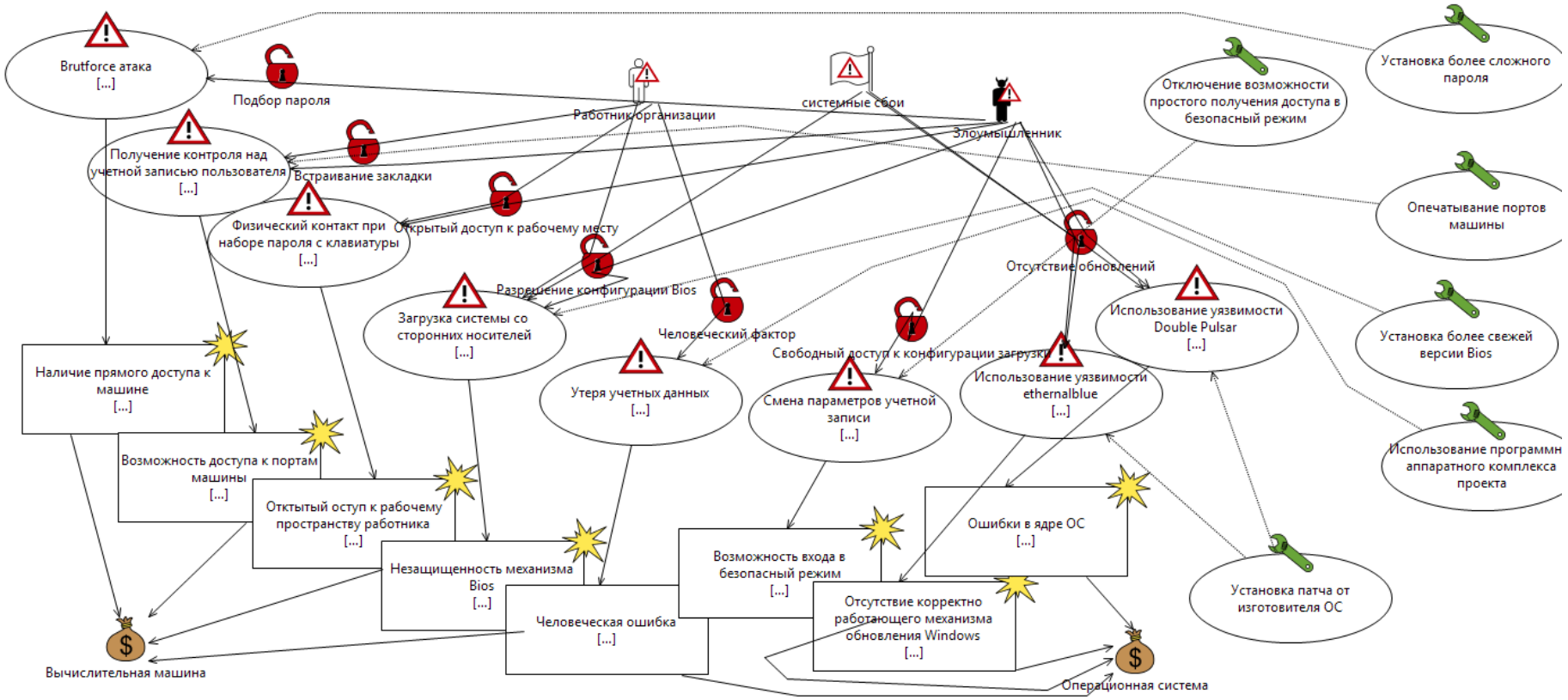

Рисунок 4.5 – Включение мер в схему с угрозами и уязвимостями

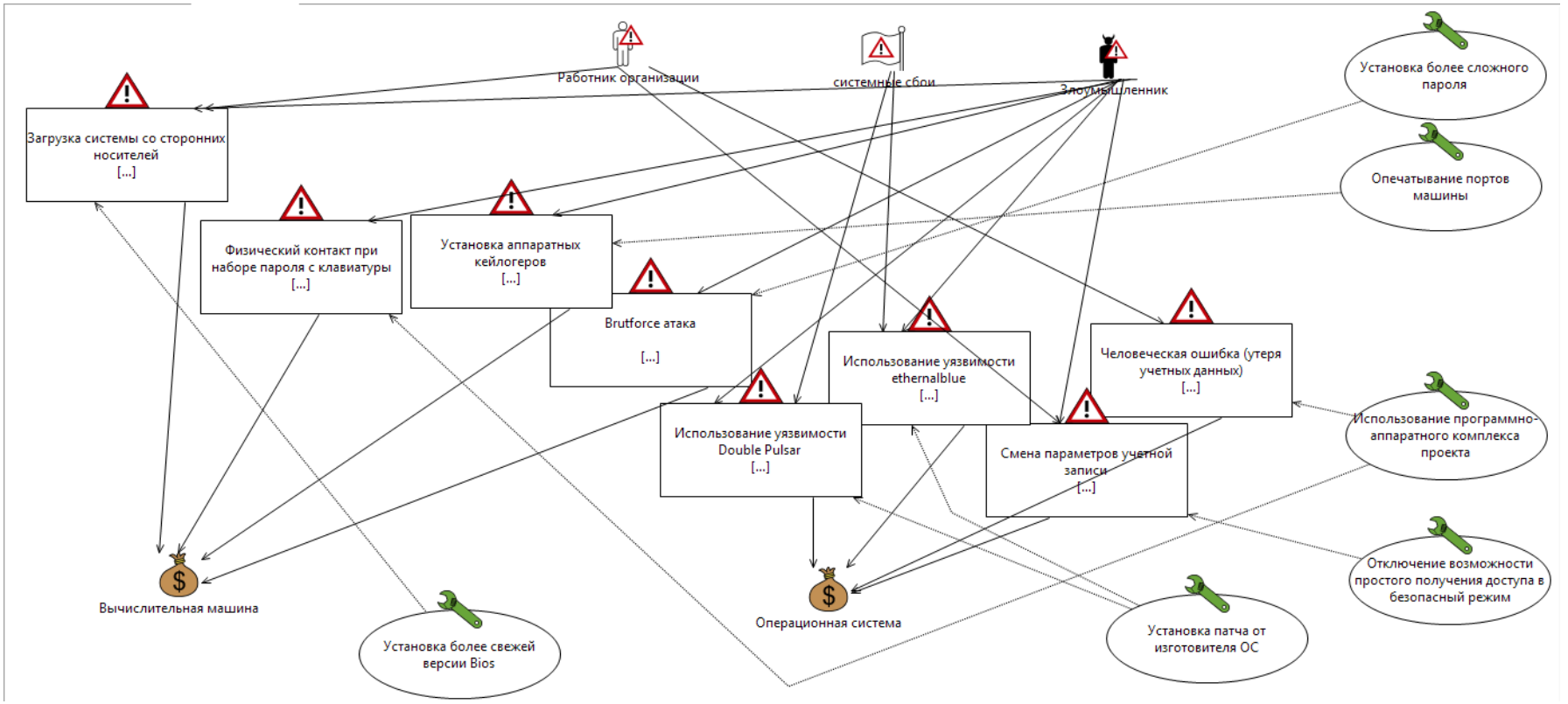

Рисунок 4.6 – Диаграмма рисков с мерами по их исключению

В ходе проведения расчетов получена итоговая таблица расчетов, показанная ниже.

Таблица 4.1 – Таблица рисков.

Код	**Угрозы**	**Уязвимости**	**Максимальный уровень риска**	**Меры по обработке риска**	**Остаточный уровень риска**	**Дата**	**Комментарии**
А. Вычислительная машина							
A.1	Brutforce атака	Наличие прямого доступа к машине	32	Установка более сложного пароля	12	27.06.2020	Достаточно важный аспект, легкореализуемый при встраивании меры из угрозы A.3
A.2	Получение контроля над учетной записью пользователя	Возможность доступа к портам машины	12	Опечатывание портов машины	12	28.06.2020	Желательно произвести в ближайшее время, вместе с операцией блокировки на контроллере домена
A.3	Физический контакт при наборе пароля с клавиатуры	Откртытый оступ к рабочему пространству работника	48	Использование программно-аппаратного комплекса проекта	16	29.06.2020	Провести курсы обучения по работе с комплексом всех работников
A.4	Загрузка системы со сторонних носителей	Незащищенность механизма Bios	24	Установка более свежей версии Bios	8	30.06.2020	Реализация требует заказ услуг сторонних организаций
B. Операционная система							
B.1	Утеря учетных данных	Человеческая ошибка	48	Использование программно-аппаратного комплекса проекта	12	30.06.2020	Обеспечить сотрудников рабочими смартфонами с поддержкой технологии NFC

B.2	Смена параметров учетной записи	Возможность входа в безопасный режим	18	Отключение возможности простого получения доступа в безопасный режим	3	01.07. 2020	Изучение офиуиальной документации для решения этого вопроса
B.3	Использование уязвимости ethernalblue	Отсутствие корректно работающего механизма обновления Windows	18	Установка патча от изготовителя ОС	3	02.07. 2020	Проверить текущии версии патчей, установленных на системах пользователей
B.4	Использование уязвимости Double Pulsar	Ошибки в ядре ОС	48		12	03.07. 2020	

Как известно не существует идеальных решений для предотвращения рисковых ситуаций в организации. Поэтому используя наш проект как единый актив организации произведем расчет рисков, использую описанные выше методики.

Таблица 4.2 – Таблица рисков.

Код	**Угрозы**	**Уязвимости**	**Максимальный уровень риска**	**Меры по обработке риска**	**Остаточный уровень риска**	**Дата**	**Комментарии**
А. Устройство «НТМА v1.0»							
A.1	Анализ работы механизма устройства	Использования распространенных компонентов и открытой архитектуры	36	Перевод проекта на закрытую архитектуру	27	27.06. 2020	Рассмотреть возможные варианты построения проекта исключительно на SMD компонентах
A.2	Подмена устройства	Свободный доступ к рабочему месту	81	Встраивание устройства в состав машины	27	28.06. 2020	Реализовать вариант проекта, который легко физически встроить в механизм машины пользователя

B. Программное обеспечение для Windows							
B.1	Анализ кода приложения	Отсутствие механизмов обфускации программного кода	288	Использование методов обфускации	72	30.06.2020	Включить наиболее подходящие библиотеки в структуру проекта, которые позволят реализовать механизм обфускации и шифрования конфигурационных файлов
B.2	Изучение файла базы пользователей	Отсутствие механизма шифрования файлов конфигурации приложения	108	Установка механизма шифрования файлов конфигурации	18	27.06.2020	
B.3	Анализ файла подключения		108		18	28.06.2020	
C. Программное обеспечения для Android							
C.1	Анализ кода приложения	Отсутствие механизмов обфускации программного кода	36	Использование методов обфускации	10	30.06.2020	Реализация механизма обфускации для приложения на Android не так важно, поскольку в работе проекта не происходит никаких взаимодействий конфиденциальных данных, однако можно использовать авторство и проверку подлинности приложения, для исключения возможности подмены

Вывод

По итогам данной главы были проведены расчеты рисков возникающих в организации относительно активов на риски которых может повлиять введение в эксплуотацию программно-аппаратного комплекса, разрабатываемого в дипломном проекте. По итогам получилось, что введение в организацию проекта позволяет снизить средний показатель риска в 3 раза. Благодаря проекту удалось понизить риски свзянанные с Bruteforce атаками на учетные записи пользователей за счет усложнения парольной комбинауии.

Поскольку проект позволяет исключить человеческий фактор, который связан с забыванием довольно сложных паролей, администраторам организации будет гораздо проще увеличивать криптостойкость паролей таким методом. Помимо этого, проект позволяет снизить риски связанные с физическим контактом пользователя и злоумышленника при наборе пароля. Набор кодовой комбинации не происходит на рабочем месте, а сама кодовая комбинация хранится на устройстве, которое всегда при работнике.

5 Безопасность жизнедеятельности

Рабочее место имеет важное значение для здоровья из-за множества опасностей, которые возникают во время смены. Эти опасности могут относиться к широкому спектру физических, химических и биологических типов. Разрабатываемый проект предусматривает размещение в офисных помещениях у рабочего места каждого сотрудника. Поэтому следует произвести анализ и расчеты некоторых негативных факторов касающихся безопасности жизнедеятельности работников.

5.1 Анализ потенциально опасных и вредных факторов в оффисе

Слова «риск» и «опасность» часто используются взаимозаменяемо. Однако, при рассмотрении вопросов, касающихся безопасности жизнедеятельности, важно, понимать разницу между ними. При проведении анализа потенциально опасных и вредоностных факторов существует шесть основных категорий:

Биологические опасности включают вирусы, бактерии, насекомых, животных и т. д., которые могут оказывать вредное воздействие на здоровье. Например, плесень, кровь и другие жидкости организма, вредные растения, сточные воды, пыль и паразиты.

Химическая опасность - это опасные вещества, которые могут причинить вред. Эти опасности могут привести как к ухудшению здоровья, так и к летальным исходам, таким как раздражение кожи, раздражение дыхательной системы, слепота.

Физические опасности - это факторы окружающей среды, которые могут нанести вред работнику без необходимости контакта с ними, включая высоту, шум, радиацию и давление.

Опасности связанные с безопасностью, которые создают небезопасные условия труда. Например, оголенные провода или поврежденный ковер могут привести к физическим травмам. Иногда они включаются в категорию физических опасностей.

Эргономические опасности являются результатом физических факторов , которые могут привести к травмам опорно - двигательного аппарата. Например, плохая настройка рабочего места в офисе, плохая осанка и неверное расположение работника в помещении.

Психосоциальные опасности – опасности к которым относятся факторы неблагоприятного воздействия на психическое здоровье. Например, сексуальные домогательства, стресс и насилие на рабочем месте.

Некоторые из вышеперечисленных категорий встречаются достаточно редко в реалиях офиса, однако некоторые их них настолько распространены, что требуют уделения огромного внимания. К примеру говоря о качестве воздуха в помещении мы подразумеваем не только биологическую опасность,

которая может возникнуть в результат некачественно спроектироанной или не работающей вентиляции. Влекущее за собой понижения приемлимой нормы концентрации кислорода в воздухе, но и как вариант более быстрому распространнию вирусов. Все это включает в сабя и психосоциальные опасности, поскольку снижение приемлимого количества кислорода влияет и на продуктивность мозговой деятельности человека.

Согласно СНиП 204.05-91 допустимая концентрация углекислого газа составляет 1,0 л/м^3. Нормализация газового состава воздуха в помещении осуществляют организацией притока наружного воздуха. Подача на одного человека должна составлять не менее 40 м^3/ч.

Помимо нарушений связанных с организацией правильной подачи воздуха, наиболее часто встречаются ошибки в проектировании искусственного освещения. Поскольку многие технически грамотные решения имеют высокую стоимость, многие работодатели игнорируют данный вопрос в угоду экономии бюджета. Однако организация технически неверного освещения может приводить к ухудшению рабочей деятельности работника, снижению его КПД. А если говорить о необратимых негативных последствиях, то ярким примером является ухудшение или потреря зрения.

Согласно СНиП 204.05-91[8] предъявляются следующие требования к осветительному оборудованию:

- -высокая светоотдача светильника;
- -низкое энергопотребление;
- -пульсация светового потока менее 1%;
- -индекс цветопередачи Ra >80.

Помимо вышеперечисленного в зависимости от тиа работ, проводимых в офисном помещении, предъявляются требованию по показателю освещенности рабочих поверхностей. Требования согласно СП 52.1330.2011 приведены в таблице ниже:

Таблица 5.1 – Требования к освещенности

Освещаемые объекты	Показатель освещенности (лк)
Кабинеты и рабочие комнаты, офисы	300
Проектные залы и комнаты, конструкторские, чертежные бюро	500
Помещения для посетителей, экспедиции	300
Компьютерные залы	400
Конференц-залы, залы заседаний	200

5.2 Расчет комфортных условий труда

5.2.1 Расчет эффективности вентиляции помещения

Для обеспечения надлежащих условий работы труда по СНиП объем производстенного помещения на каждого работающего должен быть не менее 40 м^3. Исходя из этих данных произведем расчеты необходимого и фактического показателей воздухообмена помещения. Для расчета буут использоваться данные приведенные в таблице ниже:

Таблица 5.2 – Исходные данные для расчета

Характеристики помещения	Длина А = 5 м Ширина В = 8 м Высота Н = 4 м
Количество работающих	3
Размеры форточки	Длина А = 1 м Ширина В = 0,3 м
Температура в помещении	$T_з$ = 290 К $T_л$ = 301 К
Температура снаружи	$T_з$ = 262 К $T_л$ = 297 К

Первым делом произведем расчет необходимого показателя воздухообмена в помещении:

$$L_H = \grave{L} * n \tag{5.1}$$

Где n - колличество работающих в наиболие многочисленной смене;

$\grave{L}$ - постоянный воздухообмен при помощи вентиляции (как правило принимается значение 40 м^3/ч)

$$L_H = 40 * 3 = 120 \text{ м}^3/\text{ч}$$

Далее произведем расчет фактического показателя воздухообмена. Для этого требуется найти прощадь форточки по формуле:

$$F = A * B \tag{5.2}$$

Где А – длина форточки;

В – ширина форточки.

$$F = 1 * 0{,}3 = 0{,}3 \text{ м}^2$$

Также требуется вычислить объемный вес воздуха по формуле:

$$Y = 0{,}465 * \frac{\text{Р}_\text{б}}{T} \quad (5.3)$$

Где $\text{Р}_\text{б}$ – барометрическое давление, мм рт.ст., можно принять Рб=750 мм рт.ст.;

Т - температура воздуха в кельвинах.

Вычислим показатели для лета:

$$Y_{BH} = 0{,}465 * \frac{750}{301} = 1{,}16 \text{ кг/м}^3$$

$$Y_{CH} = 0{,}465 * \frac{750}{297} = 1{,}17 \text{ кг/м}^3$$

Вычислим показатели для зимы:

$$Y_{BH} = 0{,}465 * \frac{750}{290} = 1{,}2 \text{ кг/м}^3$$

$$Y_{CH} = 0{,}465 * \frac{750}{260} = 1{,}3 \text{ кг/м}^3$$

После вычислим показатель теплового потока по формуле:

$$H_2 = \left(\frac{h}{2} - b\right) * (Y_{CH} - Y_{BH}) \quad (5.4)$$

Где h - высота помещения;

b - растояние от потолка до центра форточки;

$\text{Y}_\text{СН}$ и $\text{Y}_\text{ВН}$ – соответственно объемный вес воздуха снаружи помещения и в средине его,кгс/м3.

Для лета:

$$H_2 = \left(\frac{4}{2} - 1{,}2\right) * (1{,}17 - 1{,}16) = 0{,}008$$

Для зимы:

$$H_2 = \left(\frac{4}{2} - 1{,}2\right) * (1{,}3 - 1{,}2) = 0{,}08$$

Далее определим показатели скорости выхода воздуха по формуле:

$$V = \sqrt{\frac{2g * \Delta H_2}{Y_{ВН}}} \tag{5.5}$$

Где g - ускорение свободного падения, 9,8 м/с.

Для лета:

$$V = \sqrt{\frac{2 * 9{,}8 * 0{,}008}{1{,}16}} = 0{,}4 \text{ м/с}$$

Для зимы:

$$V = \sqrt{\frac{2 * 9{,}8 * 0{,}08}{1{,}2}} = 1{,}14 \text{ м/с}$$

По формуле фактического показателя воздухообмена вычислим показатель:

$$L_Ф = m * F * V * 3600 \tag{5.6}$$

Где m - коэффициент использования воздуха, принимает значение в рамках 0,3-0,8(как правило в расчетах принимают среднее значение 0,55);

F - площадь форточки или выходного отверстия, через которое будет выходить воздух, м^2;

V - скорость выхода воздуха через форточку или вентиляционный канал, м/с. Ее можно расчитать по формуле

Для лета:

$$L_Ф = 0{,}55 * 0{,}3 * 0{,}4 * 3600 = 237{,}6 \text{ м}^3/\text{ч}$$

Для зимы:

$$L_Ф = 0{,}55 * 0{,}3 * 1{,}14 * 3600 = 677{,}6 \text{ м}^3/\text{ч}$$

Из расчетов наглядно видно, что $L_Ф > L_Н$, это означает что конструкция естественной вентиляции в помещении требует внесения изменений,

поскольку ее показатель является неэффективным. Особенно замента разница неэффективности естественной вентиляции в период зимы, когда показатель превосходит в 1,5 раза необходимые нормы.

5.2.2 Расчет искуственного освещения офиса

Организация качественного искуственного освещения является одной из главных составляющих процесса обеспечения качественных работ на производстве. Поскольку работы производимые в данном помещении относятся к 3 классу (Высокая точность), требуется номинальное освещение в показателе 500 лк. Для проверки качества произведем расчеты по огранизации искусственного освещения и сравним с фактической обстановкой в помещении.

Производимые расчеты пологаются на харктеристи помещения и требования к системе изложенные в таблице ниже:

Таблица 5.3 – Исходные данные для расчета.

Характеристики помещения	Длина А = 5 м Ширина В = 8 м Высота Н = 4 м Высота раб.поверхности $h_{рп}$ = 0,7 м
Требование к освещенности	Ен = 500 лк
Коэффициенты отражения	$R_с$ = 50 % $R_п$ = 70 %
Коэффициент запаса	k =1,5
Коэффициент неравномерности	Z = 1,1
Тип светильников	ОДО, λ = 1,4

Приступим к расчету показателей. Для начала определим высоту светильника над рабочей поверхностью, для этого используем формулы:

Определение высоты светильника над полом

$$h_{п} = H - h_{c} \tag{5.7}$$

Где Н – высота помещения;

$h_с$ – расстояние светильников от потолка.

Определение высоты светильника над рабочей поверхностью

$$h = h_{п} - h_{рп} \tag{5.8}$$

где $h_{рп}$ – высота рабочей поверхности над полом.

$$h_{\text{п}} = 4 - 0{,}05 = 3{,}95 \text{ м}$$

$$h = 3{,}95 - 0{,}7 = 3{,}25 \text{ м}$$

После определения высоты светильника над рабочей областью, нужно рассчитать расстояние между светильниками:

$$L = \lambda * \text{h}$$
$$L = 1{,}4 * 3{,}25 = 4{,}55 \text{ м} \quad (5.9)$$

Исходя из полученных расчетов, светильники будут установлены в один ряд. Количество светильников 2 типа ОДО мощностью 40 Вт. Учитывая что в каждом светильнике данного типа установлени по две лампы, получаем общее число ламп N = 8.

Далее определим индекс помещения по формуле:

$$i = \frac{S}{h * (A + B)} \quad (5.10)$$

Где S – площадь освещаемого помещения;

A – длина помещения;

B – ширина помещения;

$$S = 5 * 8 = 40^2$$

$$i = \frac{40}{3{,}25 * (5 + 8)} = 0{,}9$$

Исходя из данных предыдущего расчета и вводных условий получаем коэффициент использования светового потока:

$$\eta = 0{,}46$$

Определим потребный световой поток ламп по формуле:

$$\Phi = \frac{\text{E} * S * \text{k} * Z}{N * \eta} \quad (5.11)$$

Где E – нормируемая минимальная освещённость;

k – коэффициент запаса, учитывающий загрязнение светильника

Z – коэффициент неравномерности освещения;

N – число ламп в помещении;
η – коэффициент использования светового потока.

$$Ф = \frac{500 * 40 * 1{,}5 * 1{,}1}{8 * 0{,}46} = 8967 \text{ лм}$$

Финальным расчет заключается в подсчете мощности искусственного освещения:

$$P = N * W \tag{5.12}$$

Где N – число ламп в помещении;
W – мощьность одной лампы светильника.

$$P = 8 * 40 = 320 \text{ Вт}$$

По итогу проделанных работы выявлено несоответствие с реальным, установленным освещением в помещении офиса. На практике используется 2 лимпы накаливания на всю площадь помещения, однако как показали расчеты, по предъявляемым к помещению требованиям в регламенте точности работ требуется установка как минимум люминисцентных ламп в светильниках типа ОДО. Поэтому требуется произвести реконструкцию помещения с изменением конфигурации светильников относительно полученных расчетов. На рисунке ниже приведен пример размещения освещения в помещении.

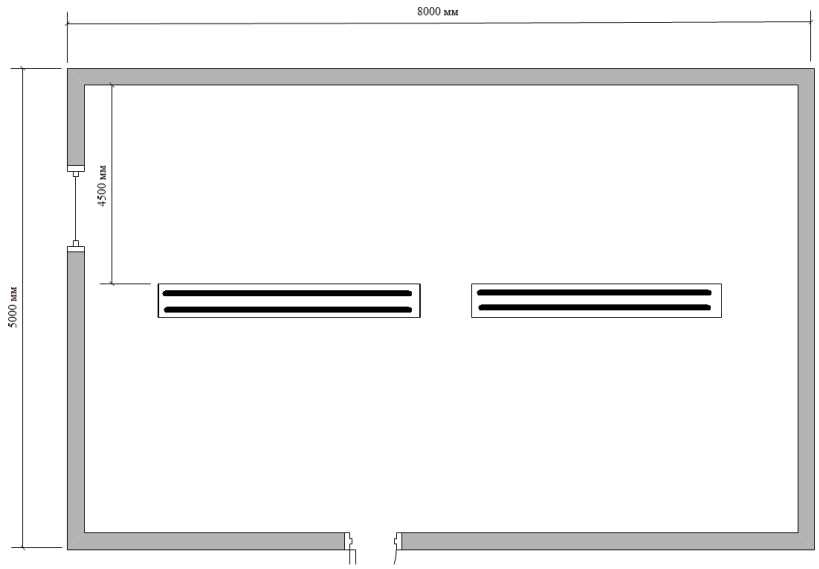

Рисунок 5.1 – Итоговый план размещения освещения

Вывод

По итогам данной главы дипломного поекта был проведен анализ всевозможных потенциально опасных и вредных факторов в помещении офиса. Рассмотрены основные причины их возникновения и методы борьбы. Помимо этого были произведены расчеты по анализу эффективности природной вентиляции в помещении, которые выявили факт неэффективности спроектированной конструкции. Также были произведены расчеты по определению и размещению искусственного освещения для помещения, которые аналогично первым, показали, что имеющиеся возможности нынешнего искусственного освещения далеки от требуемых, относительно работ, которые производятся в помещении.

Заключение

В процессе работы над дипломным проектом был изучен материал по механизму работы технологий беспроводной передачи данных RFID и NFC. Относительно поставленной задачи проекта: был изучен формат обмена данными, который входит в базовый набор функций технологии NFC. По итогам изученного теоретического материала было высказано предположение о возможности взаиодействия между двумя устройствами симбиоз которых выполнил бы основную цель дипломного проекта.

Для создания решения задачи разработки программных средств были выбраны наиболее оптимальные и свежие среды разработки с достаточно обширной базой информации, так или иначе касающейся специфики дипломного проекта. Помимо программного обеспечения, специфика диплома подразумевает разработку аппаратной части. Инструменты, используемые для решения задачи были определены исходя из степени открытости исходного кода библиотек, применяемые при разработке, а также доступности ресурсов для студента. Помимо сред разработки были рассмотрены технологии, которые могут обеспечить совместную работу разработанных программ и устройства.

По итогам проделанных работ был разработан программно-аппаратный комплекс, реализующий альтернативную работу механизма аутентификации пользователя в операционной системе Windows. Готовый проект позволяет увеличить скорость аутентификации пользователя в системе, повышает возможность усиления криптостойкости основного пароля пользователя за счет использования более длинных и сложных для подбора комбинаций символов. Были поставлены и успешно реализованы задачи:

- изучен теоретический материал по тематике NFC;
- реализован альтернативный вариант аутентификации, использующий технологию RFID по стандарту MifareClassic;
- построен чертеж устройства средствами EasyEDA;
- написан корректный, отказоустойчивый код для прошивки Arduino;
- разработаны сопровождающие устройство программы в средах Visual Studio (C#) и Android Studio (Java);
- проведено финальное тестирование проекта;
- посчитаны риски ИБ;
- произведены расчеты, согласно стандартам БЖД.

Исходя из результатов проделанных тестов, проведенных расчетов по своему проекту, могу отметить, что программно-аппаратный комплекс позволяет производить работу механизма альтернативной аутентификации пользователя в системе. С точки зрания безопасности, при введении такого проекта в рабочую среду, он позволяет снизить некоторые риски, связанные с безопасностью активов организации. Помимо этого, получены расчетные

показатели, относительно безопасности жизнедеятельности, помогают более верно спроектировать помещение, в котором будет происходить работа.

Список литературы

1. Tom Igoe, Don Coleman and Brian Jepson «Beginning NFC» - January 2014.

2. Anne-Marie Lesas and Serge Miranda «The Art and Science of NFC Programming» - January 2017.

3. Daniel Sauter «Rapid Android Development» - April 2013.

4. Matthew Leibowitz «Xamarin Mobile Development for Android Cookbook» - November 2015.

5. Syed A. Ahson and Mohammad Ilyas «Near Field Communications Handbook» - April 2016.

6. Harlan Carvey «Windows Registry Forensics, 2nd Edition» - March 2016.

7. BS 7799-1 «Практические правила управления информационной безопасностью».

8. СНиП РК 2.04–05–2002. Естественное были и искусственное ядру освещение. Комитет всех по делам цена строительства ядро министерства всем индустрии тому и торговли было РК.– Астана, 2002.

9. Дюсебаев тому М.К. «Безопасность всех жизнедеятельности. Методические указания к выполнению в дипломных проектах». – Алматы: АИЭС, 2005.

Приложение А

```
#include "SPI.h"
#include "PN532_SPI.h"
#include "snep.h"
#include "NdefMessage.h"
#include <NfcAdapter.h>
int str = 6;
int frst = 7;
int sec = 8;
int ok = 9;
PN532_SPI pn532spi(SPI, 10);
SNEP nfc(pn532spi);
uint8_t ndefBuf[128];
void setup() {
   pinMode(str, OUTPUT);
   pinMode(frst, OUTPUT);
   pinMode(sec, OUTPUT);
   pinMode(ok, OUTPUT);
   Serial.begin(9600);
   ledload();
}
void loop() {
  //индикация запуска устройства
   ledrs();
   //Задание переменной message, храняшей NDEF сообщение
   NdefMessage message = NdefMessage();
   message.addTextRecord("Start te Ulocker!");
   //проверка на корректность установленного значения переменной
message
   int messageSize = message.getEncodedSize();
   if (messageSize > sizeof(ndefBuf)) {
      leder();
      //Запуск авгоритма трансляции сообщения на смартфон
      while (1) {
      }
   }
   //проверка на корректность установленного соединения
   message.encode(ndefBuf);
   if (0 >= nfc.write(ndefBuf, messageSize)) {
      //Индикация ошибки
      leder();
   } else {
      //Индикация перехода на второй этап
```

```
            ledok();
            digitalWrite(frst,HIGH);
            //Трансляция на COM порт первой части комбинации символов
            Serial.print(">");
            Serial.print("PAS:");
          //Режим прослушивания
         int msgSize = nfc.read(ndefBuf, sizeof(ndefBuf));
         //Проверка корректности соединения
         if (msgSize > 0) {
            //Формирования массива данных полученного сообщения
            NdefMessage msg  = NdefMessage(ndefBuf, msgSize);
            //вывод части payload из сообщения ndef
            NdefRecord record = msg.getRecord(0);
            int payloadLength = record.getPayloadLength();
            byte payload[payloadLength];
            record.getPayload(payload);
            String payloadAsString = "";
            //Посимвольная запись части массива, содержащей кодовую
комбинацию
            for (int c = 0; c < payloadLength; c++)
            {
              payloadAsString += (char)payload[c];
            }
            //Трансляция на COM порт второй части комбинации символов
            Serial.println(payloadAsString);
            //Индикация успешной трансляции
            ledok();
            digitalWrite(ok,HIGH);
            //Конеци цикла работы с модулем
         } else {
           //Индикация ошибки
           leder();
         }
         }
       }
       //Функция индикации загрузки устройства
       void ledload() {
        digitalWrite(str, HIGH);
        delay(50);
        digitalWrite(frst,HIGH);
        delay(50);
        digitalWrite(sec,HIGH);
```

```
  delay(100);
```
Продолжение приложения А

```
  digitalWrite(ok,HIGH);
  delay(1000);
  digitalWrite(str, LOW);
  delay(50);
  digitalWrite(frst,LOW);
  delay(50);
  digitalWrite(sec,LOW);
  delay(100);
  digitalWrite(ok,LOW);
}
//Функция индикации успешной операции
void ledok(){
  for (int i = 0; i<=3; i++){
    digitalWrite(ok,HIGH);
    delay(500);
    digitalWrite(ok,LOW);
    delay(500);
  }
}
//Функция индикации операции с ошибкой
void leder(){
  for (int i = 0; i<=3; i++){
    digitalWrite(sec,HIGH);
    delay(500);
    digitalWrite(sec,LOW);
    delay(500);
  }
}
//Функция индикации операции перезапуска
void ledrs(){
  digitalWrite(str, LOW);
  digitalWrite(frst,LOW);
  digitalWrite(sec,LOW);
  digitalWrite(ok,LOW);
  digitalWrite(str, HIGH);
}
```

Приложение Б

```
package com.example.nfcapp;
import java.io.UnsupportedEncodingException;
import java.util.Arrays;
import androidx.appcompat.app.AppCompatActivity;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.IntentFilter.MalformedMimeTypeException;
import android.nfc.NdefMessage;
import android.nfc.NdefRecord;
import android.nfc.NfcAdapter;
import android.nfc.NfcEvent;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;
public class NFCStart extends AppCompatActivity {
   public static final String MIME_TEXT_PLAIN = "text/plain";
   private static final String TAG = "NfcDemo";
   private TextView mTextView;
   private NfcAdapter mNfcAdapter;
   private TextView mTextNFC;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_nfcstart);
      String pas = " ";
      TextView text = (TextView) findViewById(R.id.code);
      text.setText(pas);
      //Часть для работы с определение NFC даптера
      mTextNFC = (TextView) findViewById(R.id.code);
      mTextView = (TextView) findViewById(R.id.code2);
      mNfcAdapter = NfcAdapter.getDefaultAdapter(this);
      Bundle arguments = getIntent().getExtras();
      if (arguments != null) {
         pas = arguments.get("passwd").toString();
         Toast.makeText(NFCStart.this, pas, Toast.LENGTH_LONG).show();
         Toast.makeText(this, "Устройство готово к разблокировке,
поднесите к считывателю", Toast.LENGTH_LONG).show();
      }
      handleIntent(getIntent());
```

Продолжение приложения Б

```
        //Создаем переменную nfc присвеиваем ей параметры адаптера по умолчанию
        NfcAdapter nfc = NfcAdapter.getDefaultAdapter(this);
        //Задаем тип ивента такой как Callback
        nfc.setNdefPushMessageCallback(new NfcAdapter.CreateNdefMessageCallback()
        {
           @Override
           public NdefMessage createNdefMessage(NfcEvent event)
           {
              //Задаем тип данных который будет передан в сообщении
              String message = mTextNFC.getText().toString();
              //Задаем содержимое сообщения
              NdefRecord uriRecord = NdefRecord.createMime("text/plain",message.getBytes());
              //Задаем параметры, которые будет возвращать ивент
              return new NdefMessage(new NdefRecord[] { uriRecord });
           }
        }, this, this);
     }
```

Приложение В

```
Работа с реестром
var exePath = AppDomain.CurrentDomain.BaseDirectory;//Задание пути к исполняемому файлу программы
var path = Path.Combine(exePath, "Register.reg");//Задание пути к файлу вносящему изменения в реестр
var proc = new Process();//Создаем переменную процесса
var pi = new ProcessStartInfo();//Создаем переменную вывода статуса о процессе
pi.UseShellExecute = true;//Отключаем вывод консоли
pi.FileName = path;//Задаем пусть к файлу, редактирующему реестр
proc.StartInfo = pi;//Передаем параметры запуска
proc.Start();//Запускаем файл вносящий изменения в реестр

Кнопка «Обновить»
if (comboBox1.Items.Count <= 0)
{comboBox1.Items.Clear();
string[] ports = SerialPort.GetPortNames();
for (int i = 0; i < ports.Length; i++)
{comboBox1.Items.Insert(i, ports[i].ToString());}}
//проверка наличия файлов
richTextBox1.Clear();
string[] Credit = Directory.GetFiles(@"C:/Windows/System32", "RFIDCredentials.txt", SearchOption.TopDirectoryOnly);
string[] Sett = Directory.GetFiles(@"C:/Windows/System32", "RFIDCredSettings.txt", SearchOption.TopDirectoryOnly);
string[] Dll_file = Directory.GetFiles(@"C:/Windows/System32", "RFIDCredentialProvider.dll", SearchOption.TopDirectoryOnly);
richTextBox1.AppendText("Файл конфигурации пользователей: \r\n");
if (Credit.Length == 1)
{richTextBox1.AppendText("Установлен "+System.IO.File.GetLastWriteTime(@"C:/Windows/System32/RFIDCredentials.txt").ToString()+"\r\n");}
Else
{richTextBox1.AppendText("Отсутствует\r\n"); }
 richTextBox1.AppendText("Файл конфигурации подключения: \r\n");
if (Sett.Length == 1)
{richTextBox1.AppendText("Установлен " + System.IO.File.GetLastWriteTime(@"C:/Windows/System32/RFIDCredSettings.txt").ToString() + "\r\n");}
else
{ richTextBox1.AppendText("Отсутствует\r\n"); }
        richTextBox1.AppendText("Файл библиотеки: \r\n");
```

Продолжение приложения В

```
        if (Dll_file.Length == 1)
        {
         richTextBox1.AppendText("Установлен " +
System.IO.File.GetCreationTime(@"C:/Windows/System32/RFIDCredentialProvid
er.dll").ToString() + "\r\n");}
        else
        { richTextBox1.AppendText("Отсутствует\r\n");}
```