

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра Радиотехники и информационной безопасности

«ДОПУЩЕН К ЗАЩИТЕ»
Заведующий кафедрой

(ученая степень, звание, Ф.И.О.)

« » 2017 г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка криптографической системы защиты данных для различных платформ

Специальность 5В100200 Системы информационной безопасности
Выполнил(а) Катаулин С.Н. (Ф.И.О.) Группа СИБ-13-1

Научный руководитель К.т.н., проф. Маргаров Л.И.
(ученая степень, звание, Ф.И.О.)
Л.И. «06» июня 2017г.
(подпись)

Рецензент: _____
(ученая степень, звание, Ф.И.О.)
« » 2017г.
(подпись)

Консультанты:

по экономической части:

К.т.н., доцент Салимбаева Р.О.
(ученая степень, звание, Ф.И.О.)
Р.О. «24» 05 2017г.
(подпись)

по безопасности жизнедеятельности:

Монченко Л.М., ст. пр. каф. БТи ЧЗ
(ученая степень, звание, Ф.И.О.)
Л.М. «29» 05 2017г.
(подпись)

по применению вычислительной техники:

ст. пр. Зубова С.А.
(ученая степень, звание, Ф.И.О.)
С.А. «7» июня 2017г.
(подпись)

Нормоконтролер: К.т.н., доцент Сейтенова Б.
(ученая степень, звание, Ф.И.О.)
Б. «04» июня 2017г.
(подпись)

Алматы 2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Факультет Радиотехники и связи

Кафедра Радиотехники и информационной безопасности

Специальность Системы информационной безопасности

ЗАДАНИЕ

на выполнение дипломного проекта

Студенту Капачиничу Санжару Нуржановичу
(Ф.И.О.)

Тема проекта Разработка криптографической системы защиты данных для различных платформ

Утверждена приказом по университету № 149 от «24» 10 2016 г.

Срок сдачи законченного проекта « 7 » июня 2017 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): цель проекта – разработать мессенджер на Android (обеспечивающий защиту данных при передаче) и программу на Windows (шифрующую и дешифрующую данные)

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта:

1. Собрать информацию о действующих алгоритмах шифрования
2. Выбрать среди разработанных программных продуктов
3. Создать программные продукты, протестировать их, провести анализ и сделать соответствующие выводы

Перечень графического материала (с точным указанием обязательных чертежей): рисунки 1.1-1.3-алгоритмы шифрования, рис. 2.1-архитектура ОС Android, рисунки 2.2-2.14-установка и настройка Android Studio, рис. 3.1-3.2-структура проекта на Android, рисунок 3.3-блок-схема шифрования в мессенджере, рис. 3.4-3.10-отладка приложения, рис. 3.11-установленное приложение, рис. 3.12-подключение объекта, рис. 3.13-3.15-тестирование приложения, рис. 3.16-3.17-проект в Microsoft Visual Studio, рис. 3.18-установленный exe-файл и библиотек, рис. 3.20-интерфейс программы, рис. 3.21-3.23-вкладка файл, инструкции, шифрования объектов, рис. 3.24-шифрование в программе, рис. 3.25-дешифрование в программе, рис. 3.26-лог операций, рис. 3.27-отчетка полей, рис. 5.1-план презентации

Основная рекомендуемая литература: 1) С.П.Томсенко, В.П.Батура. Основы криптографии для экономистов; 2) Б. Акин. Защита компьютерной информации; 3) Алферов А. Основы криптографии; 4) Ватаев Д.А., Савиных А.А., Савиных С.В., Яценко В.В. Базовые функции в теории кодирования и криптологии; 5) Давидов А.А., Бибочов В.В., Родик В.Н. Криптографическая защита информации; 6) Базис Android программирование для мобильных устройств, Голузанов А.А.; 7) Роджерс Рик, Мадбардо Джон, Меддиске Зилурд, Мейк Блейк. Android. Разработка приложений; 8) Программирование для Android. Самоучитель. Камениченко Д.

Консультации по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
Жокомика	Салимбаева Р.О.	28.01-29.01	<i>[Подпись]</i>
БЖД	Томсенко С.И.	28.04-29.05	<i>[Подпись]</i>
Основная часть	ст. преп. Зуева С.А./Маргаров И.И.	9.1.17-15.02.17	<i>[Подпись]</i>
Тренинг вех техн.	ст. преп. Зуева С.А.	3.4.17-5.5.17	<i>[Подпись]</i>

График

подготовки дипломного проекта

[illegible]

Дата выдачи задания «24» 10 2016 г.

Заведующий кафедрой

(подпись)

Сейтенова Е.Т.)
(Ф.И.О)


Научный руководитель
проекта

Рис

(подпись)

(Маргаров Т.И.
(Ф.И.О))

Задание принял к
исполнению студент


(подпись)

(Катаулин С.Н.)
(Ф.И.О)

Аннотация

Целью данного дипломного проекта является разработка системы защиты данных на платформах Android и Windows. В процессе написания проекта были использованы ОС Android и ОС Windows. Проведен анализ платформ для разработки программных продуктов, описана подробная установка сред разработки, показана реализация системы защиты, рассмотрены методы криптографической защиты данных, реализованы алгоритмы шифрования, реализована защита приложений.

В экономической части дипломного проекта я сделал расчет всех необходимых затрат, связанных с закупкой оборудования и программного обеспечения, а также был осуществлен расчет эксплуатационных расходов.

По части безопасности жизнедеятельности я произвел анализ естественного и искусственного освещения на базе одного помещения.

Андатпа

Бұл дипломдық жобаның мақсаты Android және Windows платформаларында деректер қорғау жүйесін дамыту болып табылады. Жобаны жазу барысында Windows және Android ақпараттық жүйелері қарастырылды. Бағдарламалық қамтамасыздықты дамыту үшін платформада талдау жүргізілді, әзірлеу ортасының толыққанды орнатылуы сипатталды, жүйені қорғау жүзеге асырылды, криптографиялық деректерді қорғау әдістері қарастырылды, шифрлеу алгоритмдері және қосымша қорғау жүзеге асырылды.

Дипломдық жобаның экономикалық бөлімде мен сатып алынатын жабдықтар мен бағдарламалық қамтамасыз етуге кететін шығындарға есептеу жасадым, сондай-ақ эксплуатациялық есептеу шығындары жүзеге асырылды.

Тіршілік қауіпсіздігі бөлімінде мен бір базалық бөлменің табиғи және жасанды жарық беруге талдау жүргіздім.

Annotation

The purpose of this thesis project is the development of a data protection system on the Android and Windows platforms. In the process of writing the project was considered the Android OS and Windows OS. The analysis of platforms for the development of software products is described, detailed installation of development environments is described, protection system implementation is shown, methods of cryptographic data protection are considered, encryption algorithms are implemented, application protection is implemented.

In the economic part of the diploma project, I made a calculation of all the necessary costs associated with the purchase of equipment and software, and the calculation of operating costs was carried out.

On the part of life safety, I analyzed the natural and artificial lighting on the basis of one room.

Содержание

Введение.....	7
1 Криптографические алгоритмы и платформы реализации.....	8
1.1 Шифрование данных.....	9
1.2 Симметричное шифрование.....	9
1.3 Асимметричное шифрование.....	10
1.4 Гибридное шифрование.....	11
1.5 Метод хеширования	12
1.6 Шифрование данных на Android	12
1.7 Модели и методы шифрования данных на Android.....	14
1.8 Средства шифрования данных для Microsoft Windows.....	15
1.9 Описание работы гибридного алгоритма шифрования.....	20
2 Обзор операционной системы Android	24
2.1 Архитектура операционной системы.....	25
2.2 Инструменты для разработки и отладки приложения.....	30
2.3 Компоненты приложения Android.....	40
3 Практическая реализация	45
3.1 Структура Android-приложения	45
3.2 Алгоритм работы Android-приложения.....	47
3.3 Компиляция и установка Android-приложения	48
3.4 Тестирование Android-приложения.....	52
3.5 Структура программы для Windows	54
3.6 Установка и тестирование программы для Windows	56
4 Техничко-экономическое обоснование.....	63
4.1 Характеристика проекта	63
4.2 Трудоемкость разработки ПП.....	64
4.3 Расчет затрат на разработку ПП	65
4.4 Определение возможной (договорной) цены ПП	68
4.5 Оценка социально-экономических результатов	69
5 Безопасность жизнедеятельности.....	71
5.1 Анализ условий труда при разработке мобильного приложения	71
5.2 Расчеты.....	76
Заключение	83
Список сокращений	84
Список литературы	85
Приложение А	86
Приложение Б	93

Введение

Информация в любом её виде и на любую тематику должна быть достоверной, актуальной, выверенной и неповрежденной, поскольку от этого зависят не только результаты научных исследований, но и практические решения, принимаемые на ее основе. Во многих случаях эта информация имеет конфиденциальный характер (например, когда она касается частных сведений о лице или предназначена только для служебного пользования). Тогда следующая информация должна иметь адекватную защиту.

В своей работе я рассмотрю существующие алгоритмы шифрования данных, их преимущества и недостатки, проведу сравнительный анализ алгоритмов, разработаю криптографическую систему защиты данных, которая будет вбирать в себя лучшие качества существующих алгоритмов шифрования.

Далее разработаю приложение по защите данных на устройстве с операционной системой Android, а также программный продукт на Windows, который шифрует и дешифрует данные. Мои приложения позволят пользователям защитить важную информацию при передаче ее другим абонентам.

Для достижения вышеуказанных целей необходимо выполнить следующие задачи:

- выдвинуть алгоритм работы ПО;
- собрать информацию о существующих методах защиты информации, проанализировать их;
- выбрать среды разработки программных продуктов;
- создать программные продукты, протестировать их, провести анализ и сделать соответствующие выводы;
- рассчитать экономические расходы;
- описать безопасность жизнедеятельности;
- сделать анализ, подвести итоги.

1 Криптографические алгоритмы и платформы реализации

Криптографические алгоритмы в основном бывают вычислительно-сложными, что делает такие алгоритмы устойчивыми к взлому противника. Теоретически возможно сломать такую систему, но это неосуществимо сделать это любым из известных практических средств. Поэтому эти схемы называются вычислительно-безопасными. Теоретические достижения, например, улучшения в целочисленных алгоритмах факторизации и быстрые вычислительные технологии требуют эти решения постоянно адаптироваться. Существуют информационно-теоретически безопасные схемы, которые доказуемо не могут быть разорваны даже с неограниченной вычислительной мощности-пример является одноразовый блокнот, но эти схемы сложнее реализовать.

На сегодняшний день криптография включает в себя не только шифрование, которое представляет собой процесс преобразования обычной информации (так называемой открытым текстом) в непонятный текст (так называемый зашифрованный), но и дешифрование. Дешифрование является обратной процедурой или другими словами это переход от непонятного шифротекста к обратно простому тексту. Шифр представляет собой пару алгоритмов, которые создают шифрование и дешифрование реверсивной формой. Детальная работа шифра управляется с помощью алгоритма и в каждом конкретном случае с помощью "ключа" передается информация. Ключ, как правило, короткая строка символов, которая необходима для расшифровки зашифрованного текста. Формально, "криптосистема" - это упорядоченный список элементов конечных возможных открытых текстов, конечных возможных ключей и алгоритмов шифрования и дешифрования, которые соответствуют каждому ключу. Клавиши имеют важное значение, как формально, так и в реальной практике. Шифры без ключей могут быть тривиальным образом нарушены, и поэтому бесполезны для большинства целей. Исторически сложилось, что шифры часто использовались непосредственно для шифрования или дешифрования без дополнительных процедур, таких как аутентификация или проверки целостности. Есть два вида криптосистем: симметричные и асимметричные. В симметричных системах, присутствует тот же ключ (секретный ключ), используется для шифрования и дешифрования сообщения. Работа с данными в симметричных системах происходит быстрее, чем в асимметричных системах, поскольку они обычно используют более короткую длину ключа. Асимметричные системы используют открытый ключ для шифрования сообщения и закрытый ключ для расшифровки. Использование асимметричных систем повышает безопасность связи. Примеры асимметричных систем включают RSA (Ривест-Шамира-Адлеман) и ECC (криптографии на эллиптических кривых). Симметричные модели включают в себя наиболее часто используемый AES (Advanced Encryption Standard), который заменил старый метод DES (Data Encryption Standard).

Криптография является практикой для изучения методов для безопасного общения в присутствии третьих лиц. В более общем плане, криптография - это построение и анализ протоколов, которые мешают третьим сторонам читать личные сообщения [1].

1.1 Шифрование данных

Шифрование — обратимое преобразование информации для того, чтобы скрыть её от посторонних лиц, обеспечивая, в то же время, авторизованным пользователям доступ к ней. В основном, задачей шифрования является конфиденциальность передаваемой информации. Важной особенностью любого алгоритма шифрования является использование ключа, который утверждает выбор преобразования множества возможных для этого алгоритма.

Пользователи являются авторизованными, если у них есть какой-то подлинный ключ. Вся сложность и проблема шифрования это то, как реализуется этот процесс.

В общем, шифрование состоит из двух компонентов - шифрования и дешифрования. С помощью шифрования образуются три состояния информационной безопасности:

- конфиденциальность - скрытие информации от несанкционированного доступа при передаче или хранении.
- целостность - предотвращение изменения информации при транспортировке или хранении.
- доступность - проверка подлинности источника информации и предотвращение отказов информации от отправителя к тому, что данные были отправлены им.

Идея заключается в том, что злоумышленник перехватывает зашифрованные данные и без ключа не может ни читать, ни изменять передаваемую информацию. Кроме того, в современных криптосистемах (с открытым ключом) для шифрования и дешифрования данных могут быть использованы различные ключи. Но, с развитием криптоанализа появились методы для расшифровки секретного ключа без текста. Они основаны на математическом анализе передаваемых данных.

Симметричное шифрование использует один и тот же ключ и для зашифровывания и для расшифровывания.

Асимметричное шифрование использует два разных ключа: один для зашифровывания (который также называется открытым), другой для расшифровывания (называется закрытым) [2].

1.2 Симметричное шифрование

В симметричной криптосистеме шифрование и дешифрование происходит с использованием одного и того же ключа (рисунок 1.1). Отсюда и название - симметричный. Алгоритм и выбранный заранее ключ известен обеим сторонам. Сохранение ключа в тайне является важной задачей, чтобы

создать и поддерживать безопасный канал связи. Таким образом, существует проблема передачи первичного ключа (синхронизация ключей). Кроме того, существуют методы криптоатак, позволяющие так или иначе расшифровать информацию без использования ключа или же используя перехват на стадии согласования. В целом, эти вопросы являются проблемой надежности конкретного алгоритма шифрования и являются аргументом в выборе конкретного алгоритма.



Рисунок 1.1 – Симметричное шифрование

1.3 Асимметричное шифрование (с открытым ключом)

В системе с двумя ключами используется принцип открытого и закрытого ключей, связанных с определенным математическим способом друг с другом. Открытый ключ передается по открытому (т.е. незащищенному, доступному для наблюдения) каналу и используется для шифрования сообщения, а также для проверки электронной подписи. Для того чтобы расшифровать сообщение и сгенерировать цифровую подпись используется секретный ключ.

Схема на рисунке 1.2 решает проблему симметричных схем, связанных с начальной передачей ключа другой стороне. Если злоумышленник перехватывает симметричный ключ схемы, он может как «слушать», так и вносить изменения в передаваемую информацию. В асимметричных системах на другую сторону передается открытый ключ, который позволяет шифровать, но не расшифровывать информацию. Таким образом, решается проблема симметричных систем, связанных с синхронизацией ключа.



Рисунок 1.2 – Асимметричное шифрование

1.4 Гибридное шифрование

Гибридное шифрование – это шифрование, которое включает в себя лучшие качества криптосистемы с открытым ключом с высокой скоростью шифрования симметричных криптосистем, при этом используя симметричный ключ для шифрования данных и асимметричный для шифрования самого симметричного ключа [3].

Гибридные системы в большинстве случаев работают следующим образом. Сначала генерируется ключ на один сеанс. После этого применяется симметричный алгоритм для шифрования сообщения. Если же мы столкнулись с блочным шифрованием, то необходимо использовать режим шифрования (например CBC), который даст возможность шифровать сообщение с длиной, большей длины блока. А сам случайно сгенерированный сеансовый ключ должен быть зашифрован с помощью открытого ключа получателя сообщения, а в этом этапе применяется асимметричная система шифрования (RSA или Алгоритм Диффи-Хеллмана). Так как сеансовый ключ не имеет большой длины, то его шифрование не занимает много времени. Шифровать сообщения асимметричным методом вычислительно сложнее и дольше, поэтому тут лучше воспользоваться симметричным шифрованием. После шифрования остается отправить сообщение, зашифрованное симметричным алгоритмом и соответствующий ключ в зашифрованном виде. Получатель сначала расшифровывает ключ с помощью имеющегося у него закрытого ключа, а затем с помощью расшифрованного ключа получает полное сообщение в читабельном виде. Схема гибридного шифрования представлена на рисунке 1.3.

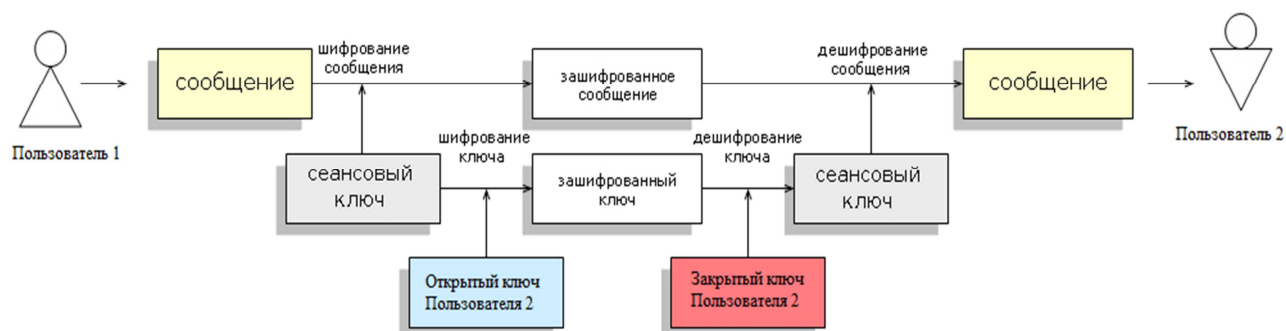


Рисунок 1.3 – Схема гибридного шифрования

1.5 Метод хеширования

При использовании кодирования по методу хеширования, создается уникальная подпись фиксированной длины для сообщения или набора данных. Хеш создается специальным алгоритмом или хеш-функцией и используется для сравнения данных. Хеш уникален для каждого набора данных или сообщения, поэтому небольшое изменение данных приведет к разительному отличию хеша, что будет свидетельствовать об отличии двух якобы одинаковых данных.

Метод хеширования отличается от других методов кодирования тем, что после кодирования хеш не может быть расшифрован или изменен. Это значит, что если злоумышленник получит хеш-код, он не сможет его декодировать и получить исходное сообщение. Распространенные методы хеширования: Message Digest 5 (MD5) и Secure Hashing Algorithm (SHA) [4].

1.6 Шифрование данных на Android

В отличие от iPhone устройства Android автоматически не шифруют данные, которые на них хранятся, даже если используется пароль для разблокировки устройства.

Шифрование устройства означает, что если устройство заблокировано, то файлы зашифрованы. Любые файлы, отправляемые и получаемые с устройства не будут зашифрованы, если не используются дополнительные методы.

Единственная разница между незашифрованным и зашифрованным устройством с точки зрения пользователя в том, что надо использовать пароль для разблокирования устройства.

Если устройство не зашифровано, то пароль — это просто блокировка экрана. Фактически, в данном случае пароль просто блокирует экран — то есть, не делает ничего, чтобы защитить файлы, которые хранятся на устройстве. Так, если атакующие находят путь обхода экрана блокировки, то они получают полный доступ к файлам.

В случае если устройство зашифровано, пароль — ключ, который дешифрует зашифрованные файлы.

То есть, когда устройство заблокировано, все данные зашифрованы и даже если атакующие найдут способ обхода экрана блокировки, то все, что они найдут, является зашифрованными данными.

Для улучшенной конфиденциальности и защиты данных обеспокоенные этим вопросом люди могут использовать шифрование информации, которая хранится на их мобильном устройстве с помощью встроенной в операционную систему «Android» функцию.

При применении этого метода следует внести ряд оговорок. Этот процесс является односторонним, то есть при его включении нет возможности его выключить без последствий, поскольку отключается механизм шифрования сбросом настроек мобильного устройства на заводские. Перед началом шифрования рекомендуется сделать резервные копии данных, и ни в коем случае нельзя прерывать процесс шифрования, иначе владельца смартфона или планшета ждут необратимые последствия, связанные с потерей информации, т. к. существует риск полностью «убить» устройство. Перед процессом шифрования следует также удостовериться в том, что установлен буквенно-цифровой пароль или PIN, который применяется для снятия блокировки экрана, так как операционная система будет использовать его в качестве ключа для дешифровки.

Шифрование данных в Android включается с помощью меню «Настройки -> Безопасность -> Зашифровать данные». При этом смартфон должен быть полностью заряжен и подключен к заряднику, а в качестве метода разблокировки использоваться PIN-код или пароль (Настройки -> Безопасность -> Блокировка экрана -> PIN-код), который следует ввести перед запуском операции шифрования. Смартфон предупредит о том, что операция займет около часа, в течение которого устройство будет несколько раз перезагружено.

Смартфон загрузит минимальную версию системы с подключением временной файловой системы к точке /data и начнет шифровать данные, выводя прогресс операции на экран. Само шифрование происходит следующим образом:

1. Сначала vold/cryptfs генерирует 128-битный мастер-ключ на основе случайных данных из /dev/urandom и с помощью этого ключа отображает раздел, содержащий каталог /data, в новое виртуальное криптоустройство, запись в которое приведет к автоматическому шифрованию данных с помощью мастер-ключа, а чтение — к расшифровке.

2. Мастер-ключ шифруется с помощью PIN-кода пользователя и помещается в конец раздела. Отныне при загрузке система будет спрашивать пользователя PIN-код, читать из раздела зашифрованный мастер-ключ, расшифровывать его с помощью PIN-кода и подключать зашифрованный раздел /data.

3. Чтобы зашифровать уже имеющиеся на разделе данные, система последовательно читает блоки данных из раздела и пишет их в криптоустройство, так что, по сути, происходит последовательная операция

«чтение блока -> шифрование -> запись обратно» до тех пор, пока не будет зашифрован весь раздел, кроме последних 16 Кб, в которых хранится мастер-ключ.

4. В конце операции смартфон перезагружается и при следующей загрузке система спрашивает PIN-код для расшифровки данных [5].

1.7 Модели и методы криптографической защиты данных на платформе Android

Каждый из нас хранит на мобильном телефоне изрядное количество конфиденциальной информации. Для кого-то это всего лишь пароли от различных сетевых сервисов, другие ответственны за хранение важной документации, третьи уже не первый год занимаются разработкой инновационной программы. В любом случае, данные необходимо беречь от посторонних лиц, что в нашем мобильном мире сделать довольно проблематично без использования систем шифрования.

Взглянув на список зашифровывающего ПО для Android, и проанализировав степень популярности и актуальности каждого из них, я пришел к выводу, что есть только четыре безопасные и поддерживаемые криптосистемы для шифрования жестких дисков и других носителей информации на лету:

1) loop-aes – модификация стандартного Linux-драйвера loop.ko, которая не только подключает устройства и образы в loopback-режиме, но и позволяет производить их шифрование на лету;

2) LUKS/dm-crypt – система шифрования, основанная на стандартной подсистеме шифрования Linux-ядра под названием dm-crypt и следующая рекомендациям TKS1/TKS2;

3) TrueCrypt – кроссплатформенная система шифрования жестких дисков и образов с графическим интерфейсом;

4) EncFS – файловая система уровня пользователя, выполняющая шифрование данных на уровне файлов, а потому способная работать поверх любой ФС.

Каждая из этих систем имеет свои преимущества и недостатки, поэтому споры о том, что именно использовать, не прекращаются до сих пор.

Если говорить о стойкости шифрования и пригодности для повседневного применения, то здесь все в порядке, по крайней мере, у первых трех претендентов. Все три системы производят шифрование на лету и находятся ниже файловой системы, поэтому взломщик не имеет ни единого шанса узнать какие-либо подробности о хранящихся внутри тома данных.

Каждая система защищена от так называемой Watermark-атаки, с помощью которой можно определить наличие в томе определенных типов файлов (dm-crypt до сих пор использует режим шифрования CBC (Cipher Block Chaining), уязвимый для этой атаки, но его легко можно изменить на устойчивый ESSIV, LRW или XTS).

Все системы могут использовать различные алгоритмы шифрования, такие как, например, AES-256, Serpent или Twofish. Для получения доступа к данным все системы позволяют использовать зашифрованный ключ, хранящийся на USB-брелке или смарт-карте.

Loop-aes – это модифицированная версия модуля loop для Linux-ядра. На стадии передачи данных от псевдоустройства к реальному, этот модуль позволяет их шифровать с минимальной долей потери производительности при чтении/записи. После размонтирования устройства, естественно, файловая система на носителе остаётся зашифрованной и недоступной для обычного монтирования диска. Как видно из названия, используется один из лучших алгоритмов шифрования в настоящее время – AES (Rijndael). Также доступны модули с использованием алгоритмов Blowfish, Serpent, Twofish. Длина ключа варьируется от 128 до 256 бит.

Основное условие, которое следует выполнить перед началом работы, это сделать резервные копии всех важных данных (особенно, если в дальнейшем будет зашифрован основной раздел).

Пересборка ядра необходима, потому что нужно убрать стандартную поддержку loop-устройства (если она включена в виде модуля или встроена в ядро) и обеспечить соответствие текущего ядра ветке исходных кодов (обычно это /usr/src/linux). Итак, для этого нужно поставить в конфигурационном файле ядра константу CONFIG_BLK_DEV_LOOP, равную n и пересобрать его.

Затем следует этап пересборки пакетов util-linux (until-linux-ng) с предварительно примененным патчем loop-AES, который можно найти в самом дистрибутиве loop-AES, либо по ссылкам (на момент написания статьи): util-linux-2.12r-20080303.diff.bz2 и util-linux-ng-2.13.1-20080303.diff.bz2 для util-linux 2.12r и util-linux-ng 2.13.1 соответственно. Программы, которые будут изменены – это swapon/swapoff (для возможности шифрования файлов подкачки), losetup (для создания зашифрованных псевдоустройств) и mount (для поддержки монтирования зашифрованных loop-устройств).

Компиляция самого loop-aes тоже очень проста, и если надо использовать не только алгоритм шифрования AES (Rijndael), но и Blowfish, Serpent и Twofish, то команде make нужно передать в качестве параметра EXTRA_CIPHERS=y. Стоит отметить, что дополнительные модули алгоритмов шифрования включены в дистрибутив loop-aes, начиная с версии 3.2a.

Драйвер loop-aes отличается простотой реализации, непревзойденной производительностью и стойкостью к взлому, однако метод его установки настолько нетривиален, что может отпугнуть даже продвинутых пользователей.

С другой стороны, LUKS/dm-crypt опирается на стандартную подсистему шифрования носителей, появившуюся еще в ядре версии 2.5 и поддерживающая десятки различных криптоалгоритмов. LUKS/dm-crypt

доступен в любой Linux-системе. В отличие от loop-aes, до сих пор страдает от некоторых неисправленных проблем и менее производителен.

Система криптозащиты дисков LUKS/dm-crypt состоит из двух основных компонентов:

1) dm-crypt – стандартная подсистема шифрования дисков Linux-ядра версии 2.6, которая опирается на подсистему Device Mapper (dm), способную отображать дисковые устройства друг на друга, и криптографическое API (Crypto API), так же предоставляемое ядром и предназначенное для выполнения различных криптографических функций;

2) LUKS (Linux Unified Key Setup) – стандарт шифрования дисковых устройств для Linux, который описывает дисковый формат для зашифрованных данных.

Благодаря LUKS производители дистрибутивов и разработчики ПО, работающие с дисковыми устройствами, получают возможность встроить в свои продукты средства для однозначного определения шифрованных дисков и работы с ними. Например, подсистема HAL, которая сегодня используется большинством дистрибутивов в качестве прослойки для работы с оборудованием, уже давно умеет определять LUKS-диски, поэтому, если в компьютер будет вставлен флеш-накопитель, зашифрованная с помощью LUKS/dm-crypt, пользователь увидит сообщение с просьбой ввести пароль, после чего флеш-накопитель будет благополучно смонтирован. Именно в этой стандартизации заключается главное достоинство LUKS/dm-crypt перед всеми остальными решениями [6].

Для создания LUKS-совместимых шифрованных дисков предназначена утилита под названием cryptsetup-LUKS, которая в отдельных дистрибутивах (например, Ubuntu) ловко замаскирована под обычную утилиту cryptsetup.

Открытая система шифрования дисков TrueCrypt появилась для систем Windows еще в 2004 году, и уже через год в нее была добавлена поддержка Linux (версия 4.0), которая стала полноценной в 2008 году, когда была выпущена TrueCrypt 5.0 с графическим интерфейсом как для Windows, так и для Linux.

TrueCrypt в первую очередь предназначена для шифрования образов ФС, но может быть использована и для шифрования целых разделов. Так же, как и LUKS, Linux-версия TrueCrypt опирается на подсистему dm-crypt, но, в отличие от первой, использует fuse для монтирования зашифрованных устройств/образов. Это оставляет свой отпечаток на производительности и делает TrueCrypt более медленной в сравнении с LUKS, однако, и у нее есть свои сильные стороны.

Во-первых, TrueCrypt по-настоящему кроссплатформенна, версии ПО есть для Windows, Mac OS X и Linux, их ядро абсолютно одинаково, поэтому никаких проблем при переносе образов между системами возникнуть не может (для чтения LUKS-разделов под Windows есть программа FreeOTFE, но за ее поддержку отвечают сторонние разработчики).

Во-вторых, TrueCrypt умеет создавать скрытые зашифрованные тома внутри уже существующих томов, причем делает это так, что формально нельзя доказать их наличие.

В-третьих, TrueCrypt создает тома такими, что их невозможно отличить от 29 случайных данных, что полезно при сокрытии информации (LUKS, как было описано выше, добавляет к любому тому заголовок, по которому его легко найти).

В-четвертых, TrueCrypt позволяет менять пароли или файлы ключей для тома без потери данных (LUKS требует пересоздания тома).

TrueCrypt, пришедший в Linux из мира Windows-систем, медленнее LUKS/dm-crypt, но, в отличие от последнего, предоставляет по-настоящему кроссплатформенное решение (тома TrueCrypt могут быть прочитаны в Windows и Mac OS X), обладает встроенным графическим интерфейсом и позволяет создавать так называемые скрытые тома (невидимые зашифрованные тома внутри зашифрованных томов).

TrueCrypt совсем не уступает LUKS по степени обеспечения безопасности зашифрованных данных. В последней версии программы используется все тот же режим шифрования XTS, который использовался при создании зашифрованных разделов с помощью LUKS. В качестве алгоритмов шифрования могут быть использованы алгоритмы AES, Twofish и Serpent, ни один из них еще не был скомпрометирован. Более того, TrueCrypt позволяет использовать так называемые каскады алгоритмов, когда зашифрованный одним алгоритмом блок данных повторно шифруется другим.

Виртуальная файловая система EncFS распространяется в виде обычного дистрибутивного пакета и не требует для своей работы ничего, кроме поддержки фреймворка fuse в ядре, libfuse, OpenSSL и небольшой библиотеки для логирования.

Создать зашифрованную с помощью EncFS файловую систему очень просто, для этого подойдет любой каталог.

EncFS – самое медленное, наиболее уязвимое, но настолько притягательно простое и удобное решение, что его нельзя обойти стороной.

Если говорить о стойкости шифрования и пригодности для повседневного применения, то здесь все в порядке, по крайней мере, у первых трех претендентов. Все три системы производят шифрование на лету и находятся ниже файловой системы, поэтому взломщик не имеет ни единого шанса узнать какие-либо подробности о хранящихся внутри тома данных. В стороне от loop-aes, LUKS/dm-crypt и TrueCrypt стоит простая и с виду незамысловатая программа EncFS. В отличие от своих тяжеловесных собратьев, она работает поверх уже существующей файловой системы и поэтому раскрывает злоумышленнику кучу самой разнообразной информации, включая всю структуру каталогов файловой системы, время создания и модификации файлов, их владельца и размер. EncFS шифрует каждый файл индивидуально, поэтому скрытыми от посторонних остаются только сами данные, хранящиеся внутри файлов, и их имена. Такая

особенность делает EncFS неприменимой для хранения серьезных данных, но наделяет ее несколькими достоинствами. Файловая система может динамически расти, инкрементальные системы бэкапа будут правильно обрабатывать файлы, зашифрованные EncFS, другие виртуальные файловые системы также могут быть зашифрованы (например, можно подключить 30 curlftpfs, создать каталог, подключить к нему encfs, и все заливаемые в него данные сохранятся на сервере в зашифрованном виде).

1.8 Средства шифрования данных для Microsoft Windows

Я исследовал самые популярные программные средства для шифрования, которые применимы на ОС Microsoft. Также провел обширный мониторинг программ, а именно Secure Disk, TrueCrypt, BitLocker и VeraCrypt. Secure Disk.

Одной из программ для шифрования данных на рабочей станции (в том числе и дома) является Secure Disk. Программа позволяет зашифровать диски компьютера частично или целиком, снабжая защиту всех данных или же их части. При совершенном шифровании диска будут защищены все данные диска, включая операционную систему, все временные, системные, и другие файлы.

Применяемый алгоритм шифрования - AES 256 - симметричный алгоритм блочного шифрования (размер блока 128 бит, ключ 128/192/256 бит), принятый в качестве образца шифрования правительством США по результатам конкурса AES.

TrueCrypt.

TrueCrypt разрешает дешифровать и шифровать данные во время их записи или считывания, образовывать виртуальные зашифрованные логические диски и шифровать иные носители информации.

TrueCrypt это компьютерная программа для шифрования «на лету» (шифрование «на лету» on-the-fly encryption обозначает, что данные автоматически дешифруются или шифруются непосредственно во время их записи или считывания, не отвлекая пользователя).

С помощью TrueCrypt можно образовывать виртуальные зашифрованные логические диски и целиком шифровать область жесткого диска или иного носителя информации, например, флоппи-диска или USB флеш-памяти. Все сохраненные при помощи TrueCrypt данные целиком шифруются, охватывая имена файлов и каталогов их нельзя прочесть (расшифровать) без применения правильного пароля/ключевого файла или без правильных ключей шифрования.

Благодаря этому организации и частные лица могут хранить все свои немаловажные файлы и пароли в безопасности. Так, с помощью TrueCrypt НКО можно избежать контроля со стороны государства, засекретив свои файлы, или применить инструмент в более мирных целях например, для хранения личных данных.

TrueCrypt имеется в версиях для Windows 7/Vista/XP, Mac OS X и Linux. Возможности TrueCrypt: TrueCrypt имеет следующими возможностями:

- организация зашифрованного виртуального диска, снабжение двух уровней правдоподобного отрицания наличия зашифрованных данных, нужного в случае вынужденного открытия пароля пользователем;
- стабильность, что позволяет запускать TrueCrypt без установки в операционной системе, содействие создания зашифрованного динамического файла на дисках NTFS, шифрование системного логического либо физического диска для Microsoft Windows-систем с до загрузочной аутентификацией;
- модификация паролей и ключевых файлов для тома без потери зашифрованных данных;
- вероятность использовать TrueCrypt на компьютере с правами рядового пользователя.

Два уровня правдоподобного отрицания. Обеспечение двух уровней правдоподобного отрицания наличия зашифрованных данных является одной из самых важных особенностей TrueCrypt. Принцип ее действия состоит в вероятности произведения зашифрованного диска с двумя паролями, по настоящему паролю доступны истинные данные на диске, а по второму иные данные. Так, в обстоятельстве, когда ваш компьютер изымает полиция, вы 18 можете сказать второй пароль, при этом все значительные данные, доступные по настоящему паролю, окажутся скрыты.

BitLocker.

Это стандартное средство шифрования дисков, встроенное в Microsoft Windows. Многие просто используют его, не устанавливая сторонних программ. С одной стороны, правильно. С другой стороны, код закрыт, и нет уверенности, что в нем не оставили бэкдоров для ФБР и прочих интересующихся. Шифрование диска осуществляется по алгоритму AES с длиной ключа 128 или 256 бит. Ключ при этом может храниться в Trusted Platform Module, на самом компьютере или на флеш-накопителе.

Если используется TPM, то при загрузке компьютера ключ может быть получен сразу из него или после аутентификации. Авторизоваться можно при помощи ключа на флеш-накопителе или введя PIN-код с клавиатуры. Комбинации этих методов дают множество вариантов для ограничения доступа: просто TPM, TPM и USB, TPM и PIN или все три сразу.

VeraCrypt.

Наиболее продвинутый клон TrueCrypt. У него собственный формат, хотя есть возможность работы в режиме TrueCrypt, в котором поддерживаются зашифрованные и виртуальные диски в формате «Трукрипта». В отличие от CipherShed, VeraCrypt может быть установлена на один и тот же компьютер одновременно с TrueCrypt. В TrueCrypt используется 1000 итераций при генерации ключа, которым будет зашифрован системный раздел, а VeraCrypt использует 327 661 итерацию. Для стандартных (не системных) разделов VeraCrypt использует 655 331 итерацию для хеш-

функции RIPEMD-160 и 500 000 итераций для SHA-2 и Whirlpool. Это делает зашифрованные разделы существенно более устойчивыми к атаке прямым перебором, но и значительно снижает производительность работы с таким разделом.

1.9 Описание работы гибридного алгоритма шифрования

1.9.1 Отправка запроса

Пользователь А выполняет `messages.getDhConfig` для получения параметров Диффи-Хеллмана: простого p и элемента q высокого порядка. Выполнение этого метода перед каждой новой процедурой генерации ключа имеет жизненно-важное значение. Имеет смысл кэшировать значения параметров вместе с версией, чтобы не каждый раз получать все значения. Если версия, хранящаяся на клиенте, все еще актуальна, сервер вернет конструктор `messages.dhConfigNotModified`.

Ожидается, что клиент проверит, является ли p безопасным 2048-битным простым (это означает, что p и $(p-1)/2$ простые, а $2^{2047} < p < 2^{2048}$), и g генерирует циклическую подгруппу простого порядка $(p-1)/2$, т. е. является квадратичным вычетом по $\text{mod } p$. Так как g всегда равно 2, 3, 4, 5, 6 или 7, это легко сделать, используя квадратичный закон взаимности, давая простое условие на $p \text{ mod } 4g$, а именно $p \text{ mod } 8 = 7$ для $g = 2$; $P \text{ mod } 3 = 2$ для $g = 3$; Нет дополнительных условий для $g = 4$; $P \text{ mod } 5 = 1$ или 4 для $g = 5$; $P \text{ mod } 24 = 19$ или 23 для $g = 6$; И $p \text{ mod } 7 = 3, 5$ или 6 для $g = 7$. После того как g и p были проверены клиентом, имеет смысл кэшировать результат, чтобы избежать повторения длительных вычислений в будущем. Этот кэш может совместно использоваться с тем, который используется для генерации ключей авторизации.

Если у клиента имеется неадекватный генератор случайных чисел, имеет смысл передать параметр `random_length` (`random_length > 0`), чтобы сервер генерировал свою случайную последовательность случайным образом соответствующей длины.

Важно: использование случайной последовательности сервера в ее необработанной форме может быть небезопасным. Он должен сочетаться с клиентской последовательностью, например, генерируя случайное число клиента той же длины (`** client_random **`) и используя `final_random = random XOR client_random`.

Клиент А вычисляет 2048-битное число a (используя достаточную энтропию или случайный сервер) и выполняет `messages.requestEncryption` после передачи в $g_a = \text{pow}(g, a) \text{ mod } \text{dh_prime}$.

Пользователь В получает обновление `updateEncryption` для всех связанных ключей авторизации (все авторизованные устройства) с конструктором чата `encryptedChatRequested`. Пользователю должна быть показана базовая информация о пользователе А, и ему необходимо предложить принять или отклонить запрос.

Оба клиента должны проверить, что g , g_a и g_b больше одного и меньше $p-1$. Рекомендуется проверить, что g_a и g_b находятся между $2^{2048-64}$ и $p-2^{2048-64}$.

1.9.2 Принятие запроса

После того как пользователь В подтвердит создание секретного сеанса с А в клиентском интерфейсе, клиент В также получит обновленные параметры конфигурации для метода Диффи-Хеллмана. После этого он генерирует случайное 2048-битовое число b , используя правила, аналогичные правилам для a .

Получив g_a из обновления с `encryptedChatRequested`, он может сразу же генерировать окончательный общий ключ: $key = (pow(g_a, b) \bmod dh_prime)$. Если длина ключа < 256 байт, нужно добавить несколько ведущих нулевых байтов в качестве отступов так, чтобы длина ключа составляла ровно 256 байт. Отпечаток пальца (`key_fingerprint`) равен 64 последним битам SHA1 (ключ).

Примечание: отпечаток используется в качестве проверки работоспособности процедуры обмена ключами для обнаружения ошибок при разработке клиентского программного обеспечения - он не подключен к визуализации ключа, используемой на клиентах, в качестве средства внешней аутентификации в секретных чатах. Ключевые визуализации на клиентах генерируются с использованием первых 128 бит SHA1 (ключевой ключ), за которым следуют первые 160 бит SHA256 (ключ используется, когда секретный чат обновлен до уровня 46).

Клиент В выполняет `messages.acceptEncryption` после его передачи g_b : $= pow(g, b) \bmod dh_prime$ и `key_fingerprint`.

Для всех авторизованных устройств Client В, за исключением текущего, обновления `updateEncryption` отправляются с помощью конструктора `encryptedChatDiscarded`. После этого единственным устройством, которое сможет получить доступ к секретному сеансу, является устройство В, которое выполнило вызов `message.acceptEncryption`.

Пользователю А будет отправлено обновление `updateEncryption` с помощью конструктора `encryptedChat` для ключа авторизации, который инициировал чат.

При использовании g_b из обновления клиент А может также получить ключ общего ключа $= (pow(g_b, a) \bmod dh_prime)$. Если длина ключа < 256 байт, нужно добавить несколько ведущих нулевых байтов в качестве отступов так, чтобы длина ключа составляла ровно 256 байт. Если отпечаток полученного ключа идентичен тому, который был передан в `encryptedChat`, входящие сообщения могут быть отправлены и обработаны. В противном случае необходимо выполнить `message.discardEncryption` и уведомить пользователя.

Для обеспечения конфиденциальности прошлых сообщений, пользователи начнут перекодировку после того, как ключ был использован

для дешифрования и шифрования более 100 сообщений или использовался более одной недели. Старые ключи затем безопасно отбрасываются и не могут быть восстановлены даже с доступом к новым ключам, которые в настоящее время используются.

1.9.3 Сериализация и шифрование исходящих сообщений

Создается объект TL (Trading Language) типа DecryptedMessage и содержит сообщение в виде обычного текста. Для обратной совместимости объект должен быть обернут в конструкторе decryptedMessageLayer с указанием поддерживаемого уровня (начиная с 8).

Полученная конструкция сериализуется как массив байтов с использованием общих правил TL. В результате массив заполняется сверху 4 байтами длины массива, не считая эти 4 байта.

Ключ сообщения, msg_key, вычисляется как 128 младших битов SHA1 данных, полученных на предыдущем шаге.

Байт-массив дополняется случайными данными, пока его длина не делится на 16 байтов.

Вычисляется ключ AES и вектор инициализации (ключ - это общий ключ, полученный при генерации ключа, x = 0):

```
Sha1_a = SHA1 (msg_key + substr (ключ, x, 32));  
Sha1_b = SHA1 (substr (ключ, 32 + x, 16) + msg_key + substr (ключ, 48 +  
x, 16));  
Sha1_c = SHA1 (substr (ключ, 64 + x, 32) + msg_key);  
Sha1_d = SHA1 (msg_key + substr (ключ, 96 + x, 32));  
Aes_key = substr (sha1_a, 0, 8) + substr (sha1_b, 8, 12) + substr (sha1_c, 4,  
12);  
Aes_iv = substr (sha1_a, 8, 12) + substr (sha1_b, 0, 8) + substr (sha1_c, 16,4)  
+ substr (sha1_d, 0, 8);
```

Данные шифруются с помощью 256-битного ключа, aes_key и 256-битного вектора инициализации, aes-iv, используя шифрование AES-256 с бесконечным расширением искажений (IGE). В верхней части результирующего массива байтов добавляются отпечаток ключа ключа шифрования key_fingerprint и ключ сообщения msg_key.

Зашифрованные данные вставляются в вызов API message.sendEncrypted и передаются на сервер для доставки другой стороне Secret Chat.

1.9.4 Расшифровка входящего сообщения

Вышеуказанные шаги выполняются в обратном порядке.

Когда получено зашифрованное сообщение, необходимо убедиться, что msg_key на самом деле равно 128 младшим разрядам SHA1-хеша дешифрованного сообщения.

Вывод

В первой главе дипломного проекта были рассмотрены следующие элементы безопасности:

- симметричное шифрование;
- асимметричное шифрование;
- гибридное шифрование;
- метод хеширования;
- средства шифрования данных на Android и Windows.

Основной недостаток симметричного шифрования заключается в необходимости публичной передачи ключей – "из рук в руки". На этот недостаток нельзя не обратить внимание, так как при такой системе становится практически невозможным использование симметричного шифрования с неограниченным количеством участников. В остальном же алгоритм симметричного шифрования можно считать достаточно проработанным и эффективным, с минимальным количеством недостатков, особенно на фоне асимметричного шифрования. Недостатки последнего не столь значительны, чтобы говорить о том, что алгоритм чем-то плох, но тем не менее.

Следующий недостаток асимметричного шифрования заключается в низкой скорости выполнения операций зашифровки и расшифровки, что обусловлено необходимостью обработки ресурсоемких операций. Как следствие, требования к аппаратной составляющей такой системы часто бывают неприемлемы. Другой недостаток – уже чисто теоретический, и заключается он в том, что математически криптостойкость алгоритмов асимметричного шифрования еще не доказана.

Дополнительные проблемы возникают и при защите открытых ключей от подмены, ведь достаточно просто подменить открытый ключ легального пользователя, чтобы впоследствии легко расшифровать его своим секретным ключом.

Какими бы недостатками и преимуществами ни обладало асимметричное и симметричное шифрование, необходимо отметить лишь то, что наиболее совершенные решения – это те, которые удачно сочетают в себе алгоритмы обоих видов шифрования, другими словами гибридный алгоритм, который я использую в своей работе.

1 Обзор операционной системы Android

Android — операционная система для интернет-планшетов, электронных книг, смартфонов, очков Google, наручных часов, цифровых проигрывателей, нетбуков, смартбуков, игровых приставок, телевизоров и других устройств. Android основана на ядре операционной системы Linux и собственной реализации виртуальной машины Java от корпорации Google. В самом начале создавалась компанией Android Inc., затем, которую выкупила компания-гигант Google. После чего Google возбудила создание коализации Open Handset Alliance (ОНА), который на сегодняшний день осуществляет поддержку и дальнейшее развитие платформы. ОС Android дает возможность строить Java-приложения, регулирующие элементом через созданные компанией Google библиотеки. Android Native Development Kit дает возможность экспортировать библиотеки и компоненты приложений, написанные на Си и других языках.

5 ноября 2007 года корпорация официально объявила новость о создании Open Handset Alliance (ОНА) и презентовала открытую платформу Android мобильных устройств, а 12 ноября 2007 года альянс представил людям первую версию пакета для программистов-разработчиков Android «Early Look» SDK и встроенный эмулятор Android.

23 сентября 2008 года официально вышла в свет начальная версия ОС Android и первый полновесный пакет разработчика SDK 1.0, Release 1. Со дня выпуска начальной версии платформы было не одно обновлений системы. Эти обновления, касаются исправления найденных ошибок и ввод новой функциональности в систему.

В 2009 году было репрезентовано четыре новых обновления ОС Android. Так, в феврале появилась версия 1.1 с исправлением многих ошибок. В апреле и сентябре выпустили ещё два обновления — 1.5 «Cupcake» и 1.6 «Donut». Обновление «Cupcake» предвнесло ощутимые поправки: браузер, воспроизведение и запись видео, виртуальная клавиатура и другие. В «Donut» впервые появились поддержка разных разрешений и плотности экрана и сетей CDMA. В октябре 2009 года появилась новое обновление ОС Android 2.0 «Eclair» с поддержкой нескольких аккаунтов Google, поддержкой браузером языка HTML5 и других новых фишек, а также после очередного обновления в пределах версии «Eclair» (2.1) появились «живые обои» и был изменен дизайн экрана блокировки.

В середине 2010 года Google презентовала Android версии 2.2 под названием «Froyo», а в конце второй половины 2010 года — Android 2.3 «Gingerbread». После обновления «Froyo» появилась возможность использовать мобильное устройство в качестве точки интернет соединения, использовать блокировку смартфона различными способами и другие нововведения, а обновление «Gingerbread» принесло еще улучшенное управление функцией вставки и копирования, обновленный контроль питанием и более полный контроль над приложениями.

Версия Android 4.0 под названием «Ice Cream Sandwich», появившаяся 19 октября 2011 года — первая многосторонняя платформа, предназначенная для смартфонов, и планшетов. Также обновление привнесли новый интерфейс улучшенный «Holo».

Осенью 31 октября 2013 года корпорация Google презентовала обновленную версию ОС Android 4.4, получившая название шоколадного батончика «KitKat» по соглашению с компанией производителем Nestlé. 25 июня 2014 Google представили Android L, доступный как для разработчиков, так и для пользователей смартфонов Nexus, а также некоторых других смартфонах.

15 октября 2014 года была официально представлена новая версия операционной системы Android 5.0 Lollipop. Самое основное обновление системы — это новый дизайн Material design. Также, в случае если на Android-устройстве установлен пароль или графический ключ, и если поблизости находятся часы хозяина устройства с Android Wear, то устройство автоматически разблокируется.

В конце 2014, а точнее 9 декабря корпорация Google сменила официальную среду разработки Eclipse, на Android Studio.

29 мая 2015 Google представила версию Android M. По словам Google главной целью новой операционной системы было — улучшить пользовательский опыт общения со смартфоном, сделать взаимодействие интуитивнее, понятнее и проще.

17 августа 2015 официально прозвучала информация, что Android M получила название Android 6.0 Marshmallow [7].

2.1 Архитектура операционной системы

Платформа Android представляет собой программный стек для мобильных устройств, который включает операционную систему, программное обеспечение промежуточного слоя (middleware), а также основные пользовательские приложения, входящие в состав мобильного телефона, календарь, карты, браузер, базы данных контактов, сообщений SMS и др.

Архитектура ОС Android подразделяется на:

- уровень каркаса приложений;
- уровень ядра;
- уровень приложений;
- уровень библиотек и среды выполнения.

На рисунке 2.1 изображены главные компоненты архитектуры ОС Android.

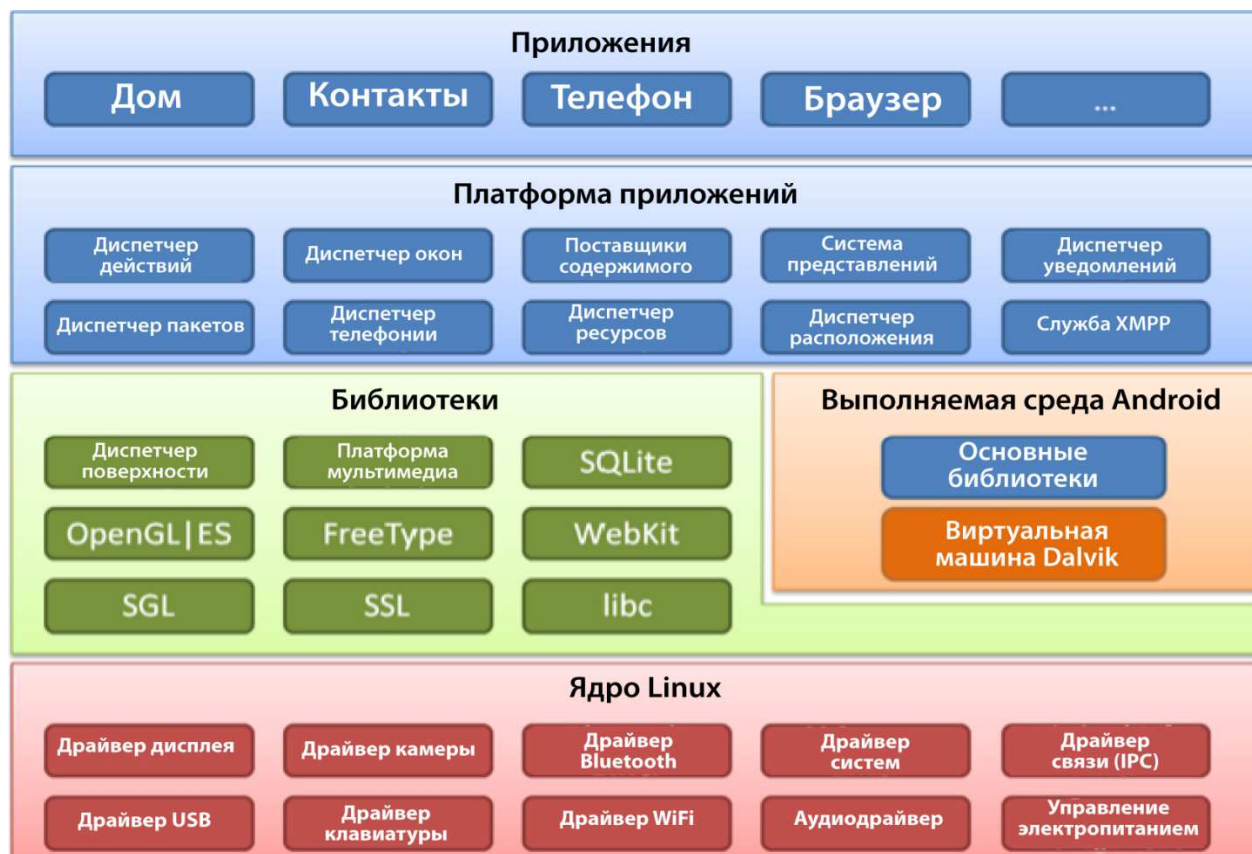


Рисунок 2.1 – Архитектура ОС Android

Основной слой. Ядро представляет собой слой абстракции между аппаратными средствами и остальной частью стека программного обеспечения. На этом уровне находятся основные услуги, такие как управление процессами, распределение памяти и управление файловой системой.

Android базируется на ядре Linux версии 2.6, но Android-система не является системой Linux в чистом виде. Android система имеет некоторые отличия, а также обеспечивает дополнительное расширение ядра Linux, которые являются специфическими для Android - ее механизмы распределения памяти, связь между процессами и т.д.

Приложения и службы могут работать в безопасных отдельных процессах, которые должны взаимодействовать друг с другом и иметь доступ к общим данным.

Android платформа поддерживает механизм IPC (Inter-Process Communication), который является основным механизмом связи между процессами. Драйвер IPC обеспечивает процессы взаимодействия, создания и обработки пулов потоков в процессах, вычисления и отображения ссылок на объекты в других процессах и синхронных запросов между процессами.

Так как Android является платформой для мобильных устройств, он должен обеспечивать экономичное потребление аккумулятора телефона,

важная роль которого выполняет система управления питанием - Android Power Management.

Она основана на стандартном драйвере управления питанием Linux, но оптимизирована для мобильных устройств с учетом их индивидуальных особенностей. Драйвер переводит систему в «спящий режим» с минимальным потреблением энергии процессора, когда не используются приложения и службы.

Программный Android-стек, разработанный с необходимой гибкостью, в том числе и для работы с целым рядом дополнительных компонентов, доступных в мобильных устройствах. Эти компоненты в значительной степени зависят от наличия определенного аппаратного обеспечения на устройстве. Они обеспечивают дополнительные функциональные возможности для мобильных устройств (сенсорный экран, камера, GPS, акселерометр, и т.д.).

На этом уровне находится набор драйверов для взаимодействия мобильного устройства с аппаратным обеспечением. Набор драйверов может отличаться в зависимости от производителя и модели устройства. По мере постоянного появления новых дополнительных техник для мобильных устройств на рынке, драйвера для них должны быть написаны на уровне ядра Linux для обеспечения поддержки оборудования, а также для настольных Linux-систем.

Преимущество использования ядра Linux в качестве основы для Android позволяет ядру верхних слоев стека программного обеспечения оставаться неизменными, несмотря на различия в используемом оборудовании. Конечно, хорошая практика программирования требует, чтобы пользовательские приложения корректно завершили свою работу в случае, если вызов ресурса не доступен, например, встроенной видекамеры, или сенсора не присутствующего в этом мобильном телефоне [8].

Уровень библиотек. Следующий уровень ядра Linux включает в себя набор библиотек C/C ++, используемый различными компонентами операционной системы. Библиотеки этого уровня можно разделить на две группы:

- системная библиотека C;
- функциональные библиотеки C/C++.

Библиотечная система основана на Berkeley Software Distribution (BSD). Корпорация Google разработала свою собственную версию системной библиотеки libc – Bionic специально для мобильных устройств на базе Linux. Это было необходимо для быстрой загрузки библиотеки в каждом процессе, и, следовательно, библиотека должна была иметь небольшой размер. Библиотека Bionic имеет размер около 200 Кб, что в два раза меньше, чем стандартные библиотеки Glibc Linux. Кроме того, было необходимо учитывать ограниченную мощность процессора мобильного устройства. Это означает, что библиотека должна быть оптимизирована для достижения максимальной производительности. Конечно, сейчас это уже не актуально, современные

мобильные устройства практически равны по мощности процессора с нетбуками, но несколько лет назад это было серьезной проблемой.

Bionic библиотека имеет встроенную поддержку для важных системных служб Android и регистрацию системных событий, но в то же время, она не поддерживает некоторые функциональные возможности, например, исключения, C++, и несовместима с GNU libc и стандартом POSIX.

Функциональные библиотеки представляют собой набор библиотек C/C++ типа:

- Surface Manager — в Android используется композитный менеджер окон, такой как Compiz (Linux), но более упрощенный. Вместо того чтобы рендериться для создания графики непосредственно в буфере дисплея, система посылает входящие команды рендеринга в закадровый буфер, где они хранятся вместе с другими, образуя своего рода композицию, а затем отображается пользователю на экране. Это позволяет системе создавать интересные эффекты бесшовные, прозрачные окна, более гладкие переходы.

- Media Framework — библиотека, реализованная на основе PacketVideo OpenCORE. С их помощью система может записывать и воспроизводить аудио и видео контента, а также выводить статические изображения. Он поддерживает множество популярных форматов, включая MPEG4, H.264, MP3, AAC, AMR, JPG, и PNG.

- SQLite — легкая и производительная реляционная база данных, используемая в Android в качестве главного движка для работы с базами данных, которые используются приложениями для хранения информации.

- 3D библиотеки - используются для оптимизирования рендеринга 3D-графики, если есть возможность, используют аппаратное ускорение. Их реализации основаны на API OpenGL ES 1.0. OpenGL ES (OpenGL for Embedded Systems) - подмножество графического программного интерфейса OpenGL, адаптированных для работы на встраиваемых системах.

- FreeType – библиотека для работы с битовыми картами, а также для растеризации шрифтов и операций над ними. Это высококачественный движок для шрифтов и отображения текста.

- LibWebCore – знаменитая библиотека шустрого браузерного движка WebKit, используемого также в настольных браузерах Google Chrome и Apple Safari.

- SGL (Skia Graphics Engine) – открытый движок для работы с 2D-графикой. Графическая библиотека обеспечивается Google и часто используется в других их программах.

- SSL - библиотеки для поддержки одноименного криптографического протокола.

- Libc – стандартная библиотека языка C, а именно её BSD-реализация, сконфигурирована для работы на устройствах на базе Linux. Называется Bionic.

Для разработчиков доступ к функциям этих библиотек реализован через использование Application Framework – каркаса приложений.

Прикладное программное обеспечение, загружаемое на мобильном устройстве, выполняет виртуальная машина Dalvik, которая, хоть и является аналогом виртуальной машины Java, сильно отличается от него. Dalvik относится к классу регистровых машин, которые идеально подходят для работы на процессорах RISC-архитектуры, которые включают в себя процессоры ARM, используемые в мобильных устройствах, в то время как стандартная виртуальная машина Java от Sun Microsystems - стековая. В результате использования регистра виртуальной машины Google рассчитывает сократить на 30 процентов количество команд по сравнению с машиной стека.

Созданные с помощью стандартного Java-компилятора class-файлы преобразуются в байт-код Dalvik (*.dex) транслятором dx, входящим в состав SDK. Внутри работающий Android выглядит как набор виртуальных машин Dalvik, каждый из которых выполняет задачу.

Виртуальная машина Dalvik, на которой построена вся операционная система корпорации Google Android, предоставляет разработчикам удобный механизм для написания приложений, которым не принципиален объем используемой памяти и мощность процессора.

Уровень структуры приложения. Уровень структуры приложения находится на вершине системных библиотек, функциональных библиотек и Dalvik VM. На этом уровне находятся основные службы Android управления жизненным циклом приложений, пакетами, ресурсами и т.д.

Программист имеет полный доступ к той же API, который используется основным приложением. Архитектура этих приложений предназначена для упрощения повторного использования компонентов. Любое разрабатываемое приложение может использовать возможности основных приложений и, следовательно, любое другое приложение стороннего производителя может использовать возможности ваших приложений (с учетом установленных разрешений). Этот же механизм позволяет повторно использовать уже разработанные компоненты.

Уровень приложений. Android мобильное устройство поставляется с набором основных приложений, включая клиентской программы электронной почты для работы с SMS, календарем, картой навигации, браузером, контактами и другими.

Интересно, что Android-платформа не делает разницы между базовыми приложениями, принадлежащих к комплекту мобильного телефона и набор программного обеспечения сторонних производителей, так что ключевые приложения, которые входят в стандартный набор программного обеспечения, могут быть заменены при желании альтернативными приложениями.

При разработке приложений программисты имеют полный доступ ко всем функциям операционной системы. Архитектура приложения разработана таким образом, что легко использовать основные компоненты, предоставляемые системой. Кроме того, можно создавать компоненты и сделать их доступными для общественного использования [9].

2.2 Инструменты для разработки и отладки приложения

Устанавливаю Java Development Kit среду разработки Android SDK и Android Studio.

Все инструменты, которые мне нужны для разработки приложений для платформы Android доступны и абсолютно бесплатны. Google предлагает для бесплатного скачивания набор библиотек и инструментов для разработки приложений – Android SDK (Software Development Kit), который предназначен для x86 компьютеров, под управлением операционными системами Mac OS, Linux и Windows XP.

Перед тем как приступить к работе над созданием Android-приложений должны быть загружены и установлены следующие программные обеспечения:

- Android SDK 24.0;
- JDK 1.8.0;
- Android Studio.

Android-приложения в настольных операционных системах запускаются в эмуляторе мобильного устройства, необходимые инструменты для разработки можно установить на любую из систем: Windows, Linux или Mac OS.

SDK включает эмулятор для всех ОС, и, поскольку Android-приложения запускаются на виртуальной машине, нет никаких преимуществ для разработки приложений в любой из этих ОС.

2.2.1 Установка Java Development Kit (JDK)

Для того, чтобы разработать программу на языке Java мне необходимо специальное ПО. Новейшие версии системного ПО, необходимого для поддержки можно загрузить с сайта компании Sun Microsystems.

Для того, чтобы начать выполнение программ необходима Java Runtime Environment (JRE). Для разработки программ также требуется набор инструментальных средств разработки программного обеспечения – Java Development Kit (JDK).

Java Development Kit – это комплект разработчика приложений на языке Java, включающий в себя компилятор Java (javac), стандартные библиотеки классов Java, примеры, документацию, различные утилиты и уже включающий в себя Java Runtime Environment. Java Development Kit доступен для бесплатной установки с сайта Sun Microsystems по адресу <http://java.sun.com/javase/downloads/index.jsp>. После загрузки JDK осуществляю установку с параметрами по умолчанию, предлагаемыми мастером установки.

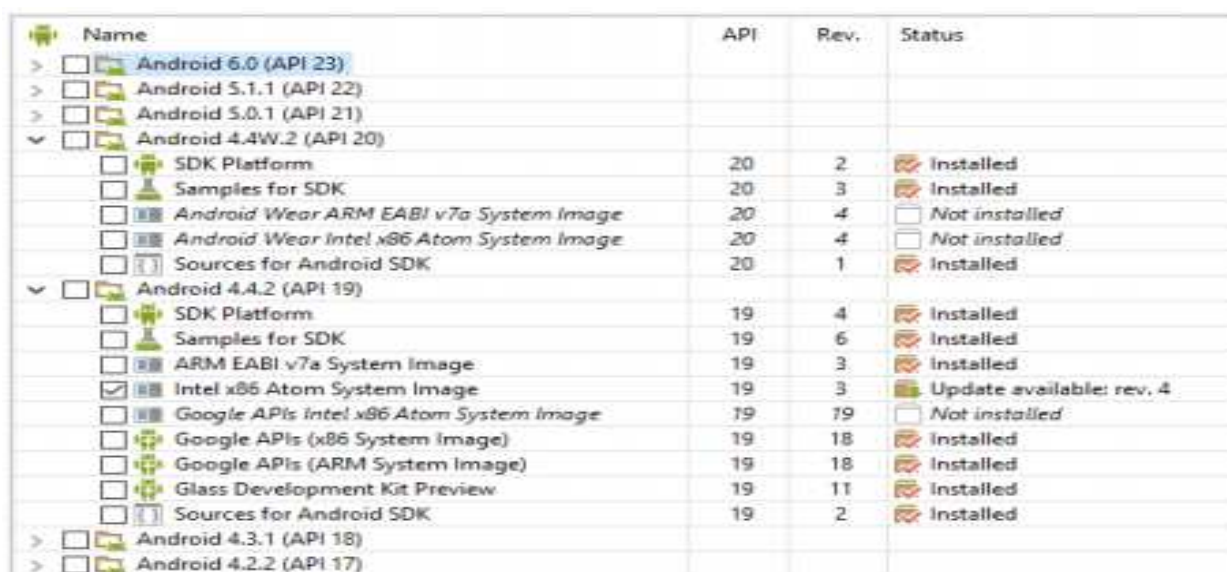
Однако в состав JDK не входит интегрированная среда разработки на Java (IDE), поэтому для разработки приложений необходимо установить Android SDK.

1.2.2 Установка Android SDK

Чтобы создавать приложения для ОС Android нужен Android SDK. SDK включает в себя эмулятор, так что нет нужды в смартфоне с ОС Android, для того, чтобы разрабатывать приложения для Android. После окончания скачивания необходимо разархивировать файл в выбранную папку. Начиная с версии 2.0 архив Android SDK включает в себя только инструментальные средства. В начальных версиях SDK архив содержал весь комплект компонентов текущей платформы Android. В версии 2.0 и выше используется специальный инструмент Android SDK и AVD Manager для установки и обновления компонентов Android SDK – библиотек, инструментов и документации.

Чтобы создавать свое приложения, нужно загрузить не меньше одной версии платформы Android, используя Android SDK и AVD Manager. Для этого необходимо подключение к Интернету, т.к. все нужные для установки и обновления компонента SDK находятся в репозитории на сервере Google.

Для того чтобы открыть AVD Manager и Android SDK, нужно запустить файл SDK Setup.exe в корневом каталоге SDK. После установки соединения с репозиторием Google в окне менеджера будет показан список доступных пакетов, как показано на рисунке 2.2.



Name	API	Rev.	Status
> <input type="checkbox"/> Android 6.0 (API 23)			
> <input type="checkbox"/> Android 5.1.1 (API 22)			
> <input type="checkbox"/> Android 5.0.1 (API 21)			
▼ <input type="checkbox"/> Android 4.4W.2 (API 20)			
<input type="checkbox"/> SDK Platform	20	2	Installed
<input type="checkbox"/> Samples for SDK	20	3	Installed
<input type="checkbox"/> Android Wear ARM EABI v7a System Image	20	4	Not installed
<input type="checkbox"/> Android Wear Intel x86 Atom System Image	20	4	Not installed
<input type="checkbox"/> Sources for Android SDK	20	1	Installed
▼ <input type="checkbox"/> Android 4.4.2 (API 19)			
<input type="checkbox"/> SDK Platform	19	4	Installed
<input type="checkbox"/> Samples for SDK	19	6	Installed
<input type="checkbox"/> ARM EABI v7a System Image	19	3	Installed
<input checked="" type="checkbox"/> Intel x86 Atom System Image	19	3	Update available: rev. 4
<input type="checkbox"/> Google APIs Intel x86 Atom System Image	19	79	Not installed
<input type="checkbox"/> Google APIs (x86 System Image)	19	18	Installed
<input type="checkbox"/> Google APIs (ARM System Image)	19	18	Installed
<input type="checkbox"/> Glass Development Kit Preview	19	11	Installed
<input type="checkbox"/> Sources for Android SDK	19	2	Installed
> <input type="checkbox"/> Android 4.3.1 (API 18)			
> <input type="checkbox"/> Android 4.2.2 (API 17)			

Рисунок 2.2 – Выбор пакетов для инсталляции

Выбрав нужные пакеты, нажимаю на кнопку Install и затем, следуя инструкциям, устанавливаю компоненты SDK. Помимо библиотек из репозитория Google, можно также дополнительно загрузить библиотеки, разработанные другими программистами (рисунок 2.3).

<input checked="" type="checkbox"/>	<input type="checkbox"/>	Extras				
<input type="checkbox"/>	<input checked="" type="checkbox"/>	GPU Debugging tools		1.0.3	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Android Support Repository		30	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Android Support Library		23.2.1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Android Auto Desktop Head Unit emulator		1.1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Play services		29	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Repository		25	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Market Apk Expansion		1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Market Licensing		1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Play APK Expansion Library		3	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Play Billing Library		5	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Play Licensing Library		2	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Android Auto API Simulators		1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google USB Driver		11	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Google Web Driver		2	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Intel x86 Emulator Accelerator (HAXM installer)		6.0.1	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Lldb		2.0.25...	<input checked="" type="checkbox"/>	Installed
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Ndk Bundle		0	<input checked="" type="checkbox"/>	Installed

Рисунок 2.3 – Установка дополнительных библиотек

После успешной установки Android SDK можно приступить к установке среды разработки Android Studio.

1.2.3 Установка Android Studio

Перед тем, как начать установку Android Studio, необходимо скачать среду разработки с официального сайта <http://developer.android.com/intl/ru/sdk/index.html> и открыть после загрузки установочный файл. На рисунке 2.4 изображено окно установки, в котором есть поле ввода, куда необходимо прописать свой путь установки или можно оставить автоматический установленный путь загрузки Android Studio, нажав кнопку Next. Если же уже установлена среда разработки Android Studio, а на компьютере установлена предыдущая версия Android Studio, необходимо поставить галочку в окошке с текстом Uninstall the previous version. На рисунке 2.5 кликаю на кнопку Next и перехожу в следующее окно установки.

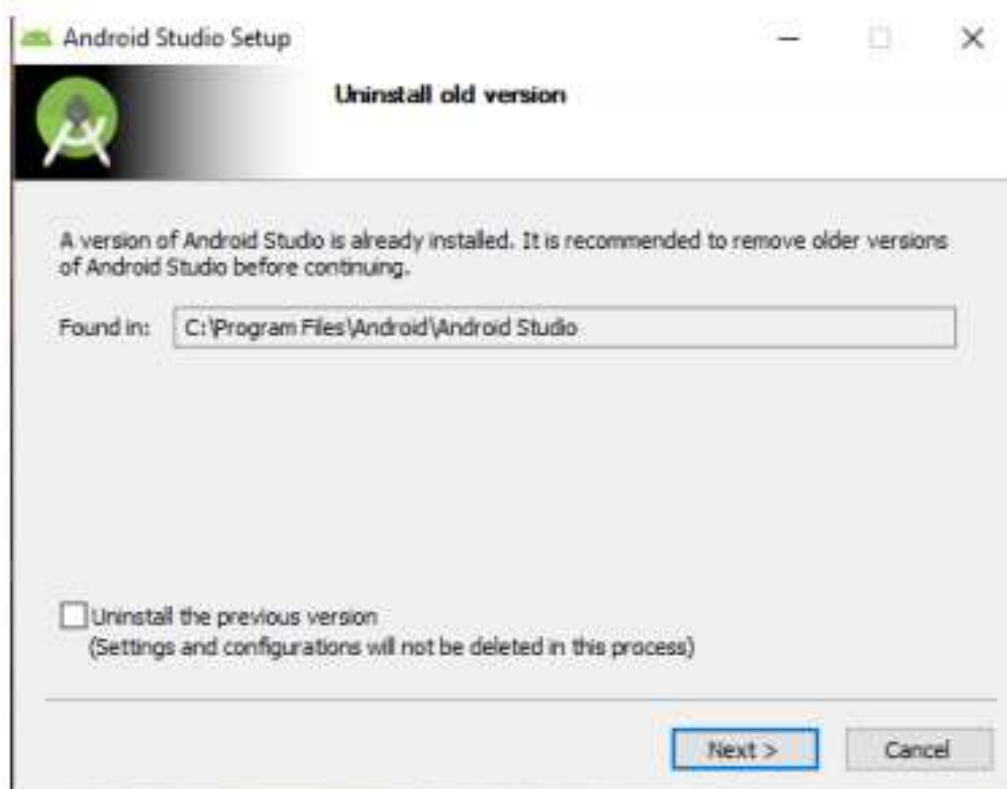


Рисунок 2.4 – Выбор пути установки Android Studio



Рисунок 2.5 – Окно приветствия установки

Плюсом является то, что Android SDK и Android Virtual Device встроены дополнительно в установку среды разработки. В случае, если уже были установлены ранее на компьютер Android SDK и AVD, можно отменить их установку, убрав галочки напротив них. Все это можно увидеть на рисунке 2.6.

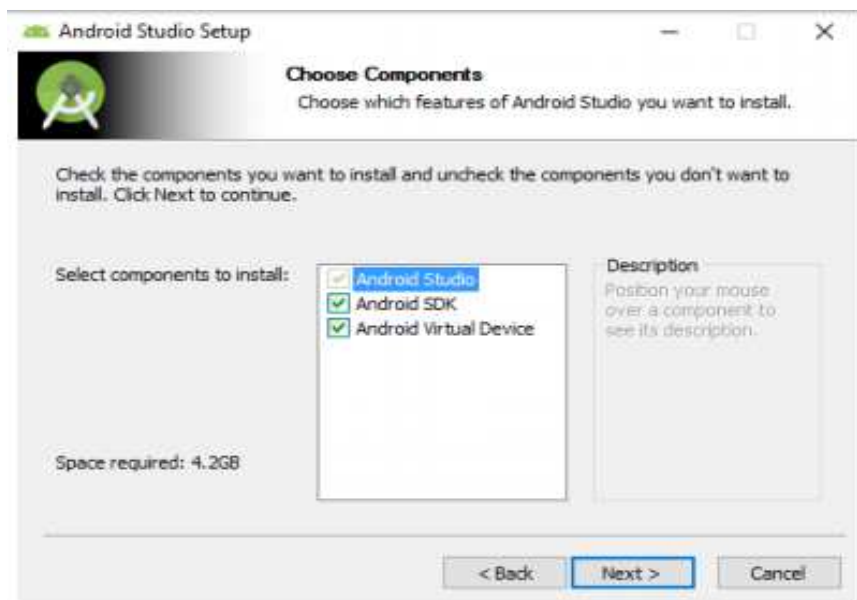


Рисунок 2.6 – Выбор компонентов установки

Необходимо прочесть и ознакомиться с соглашением на рисунке 2.7 и перейти к следующему пункту установки, нажав на кнопку “I Agree”.

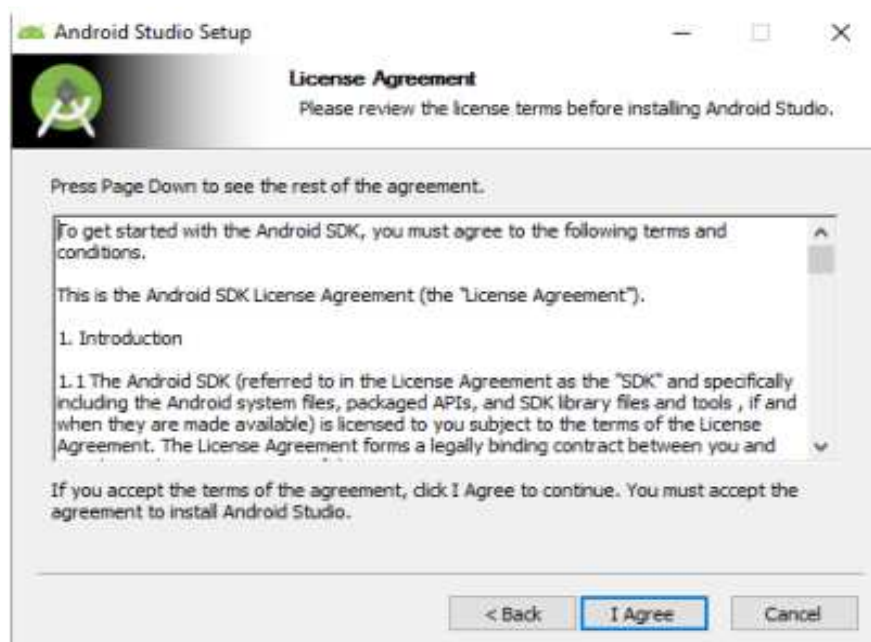


Рисунок 2.7 – Условия соглашения

В следующем окне установки, которое представлено на рисунке 2.8 можно дать любое название папке или же оставить по умолчанию название папки, в которую будет установлена среда разработки Android Studio.

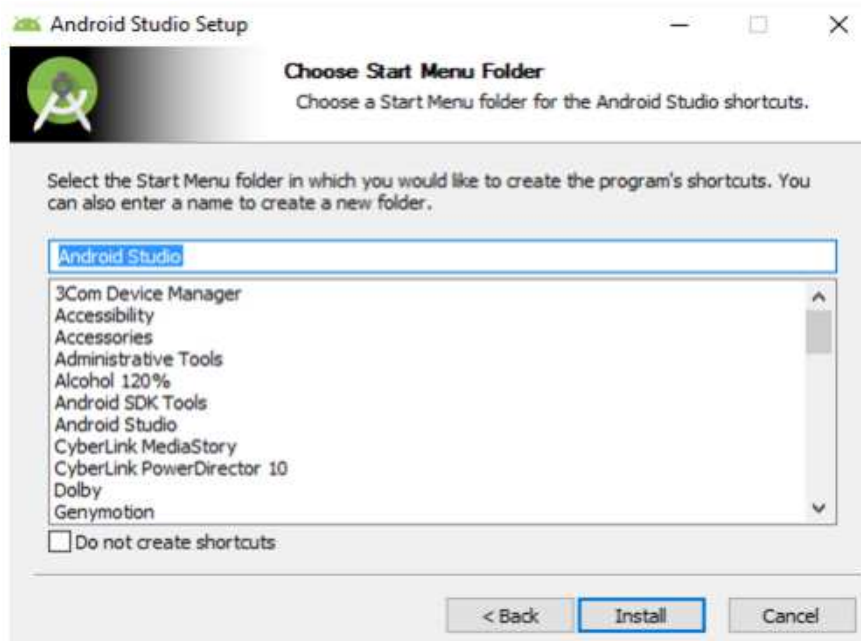


Рисунок 2.8 – Назначение названия папки

На рисунке 2.9 изображен процесс установки среды разработки Android Studio, а на рисунке 2.10 уже видна готовая к использованию после установки среда разработки.

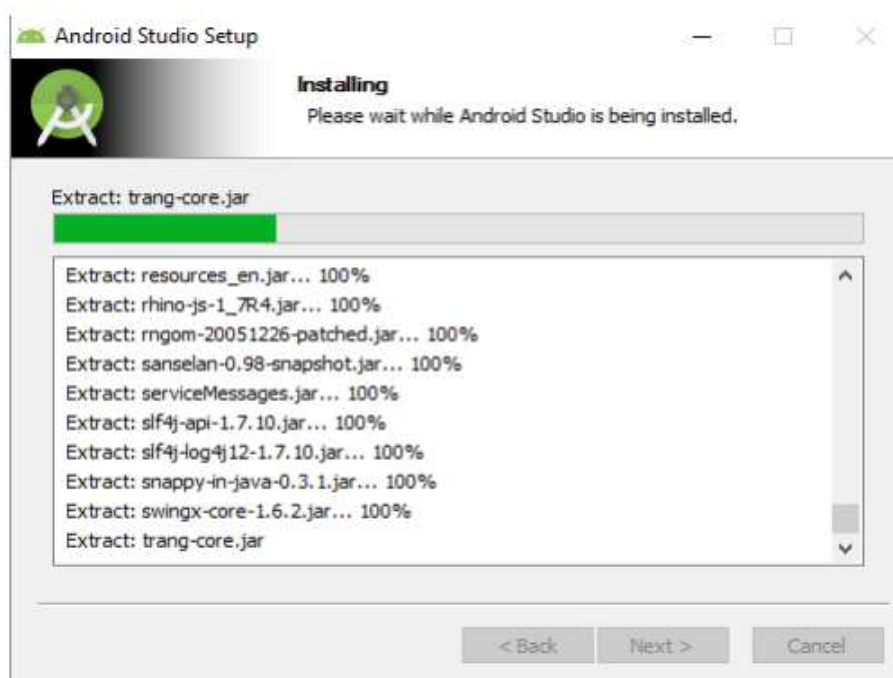


Рисунок 2.9 – Процесс установки Android Studio

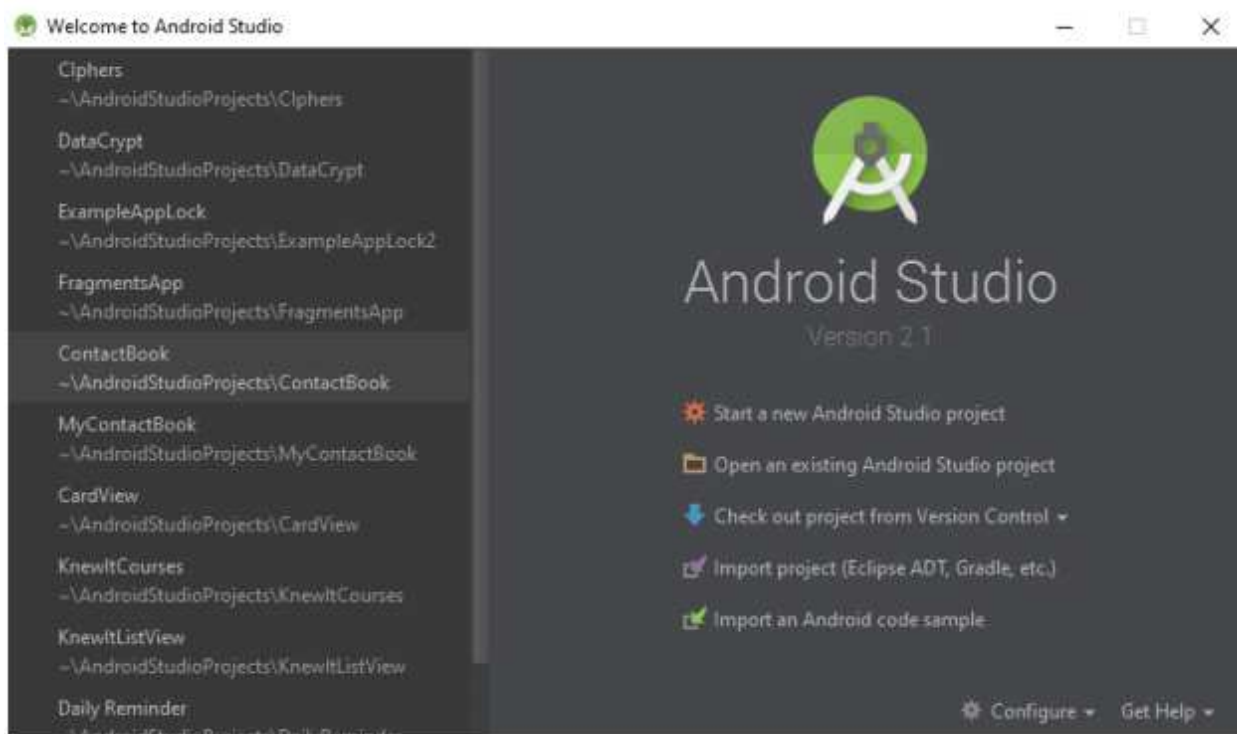


Рисунок 2.10 – Окно входа в Android Studio

Перед началом создания Android-приложения необходимо создать виртуальное устройство Android Virtual Device для тестирования приложений.

2.2.4 Создание виртуальных устройств Android (AVD) для использования в эмуляторе

Android Virtual Device (виртуальное устройство Android) – это эмулятор, который загружается на обычном компьютере. Виртуальное устройство используется для проектирования, отладки и тестирования приложений в реальной среде выполнения.

Перед тем как запускать Android-эмулятор устройства, нужно создать экземпляр Android Virtual Device (AVD). AVD определяет системное изображение и параметры настройки устройства, используемые эмулятором. Создать экземпляр AVD можно двумя способами:

- в командной строке утилитой Android, доступной в каталоге, куда я установил Android SDK, в папке tools;
- с помощью визуального инструмента Android SDK and AVD manager.

Его можно запустить из корневого каталога Android SDK или в среде Eclipse, выбрав пункт меню Window I Android SDK and AVD Manager. При этом появится окно Android SDK and AVD Manager, с помощью которого можно создавать и конфигурировать эмуляторы мобильного устройства, а также загружать обновления Android SDK (рисунок 2.11).

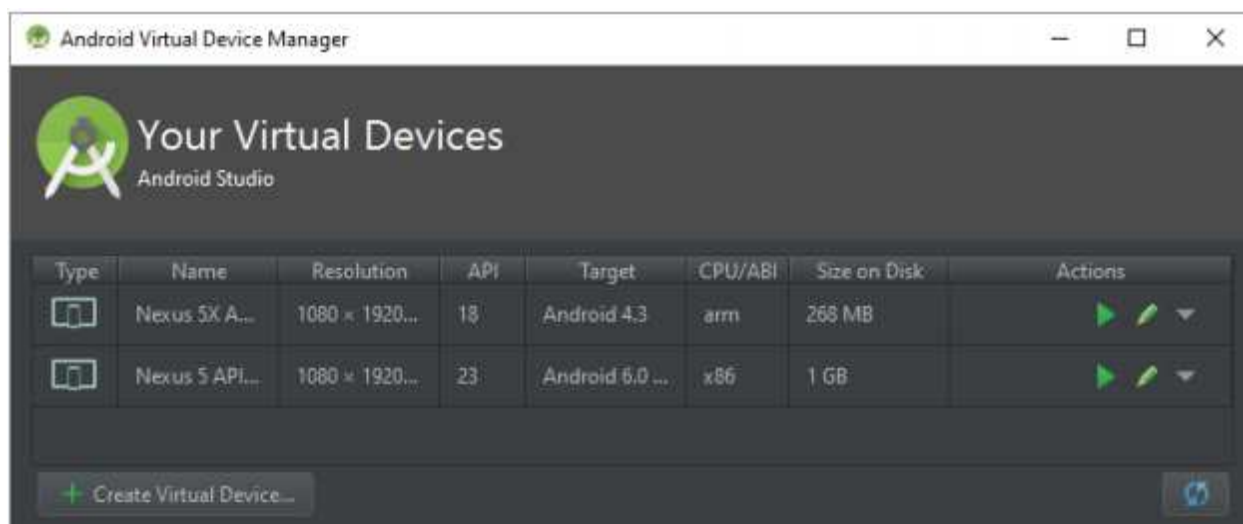


Рисунок 2.11 – Окно Android SDK and AVD Manager

Окно Android SDK and AVD Manager также появится, если в командной строке вызвать команду “android” без параметров.

В левой нижней части панели Your Virtual Devices, нажав на кнопку Create Virtual Device, откроется окно Select Hardware для выбора нового виртуального устройства (рисунок 2.12).

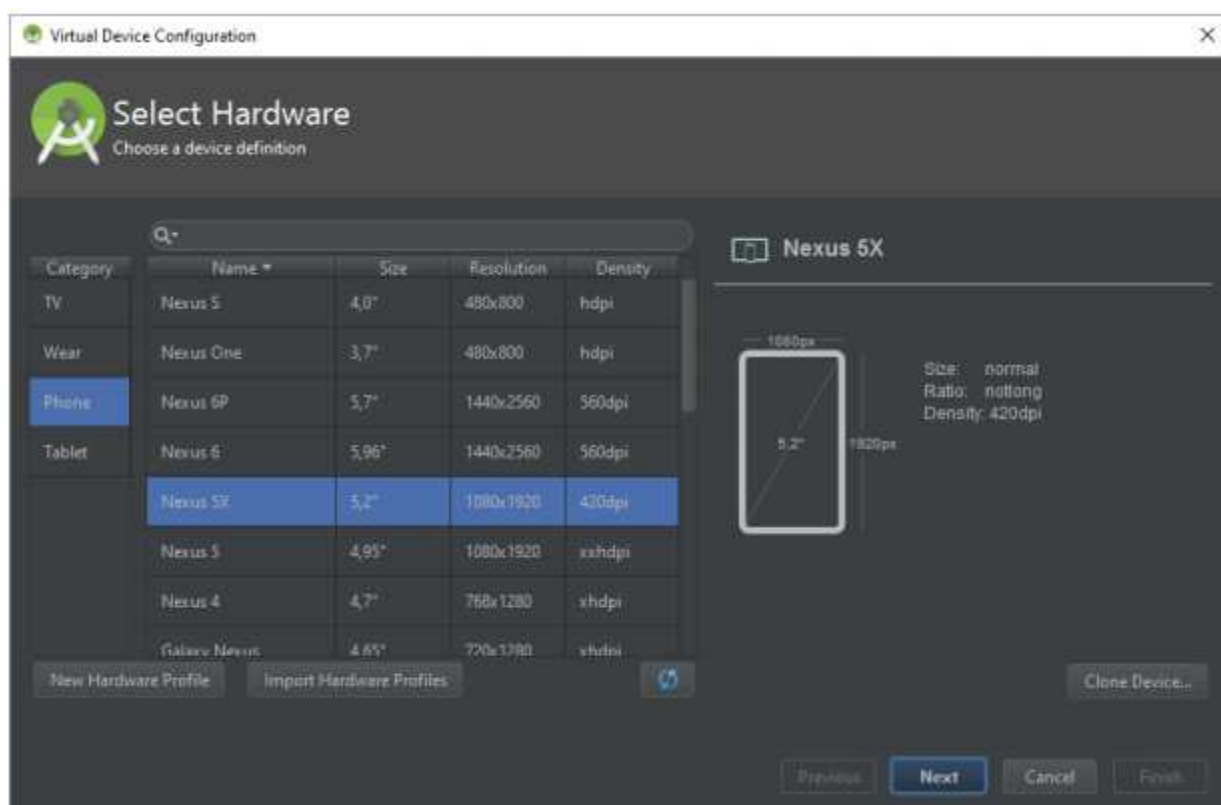


Рисунок 2.12 – Диалоговое окно выбора нового AVD

После выбора устройства и нажатия на кнопку “Next” откроется следующее окно Android Virtual Device (AVD) (рисунок 2.13).

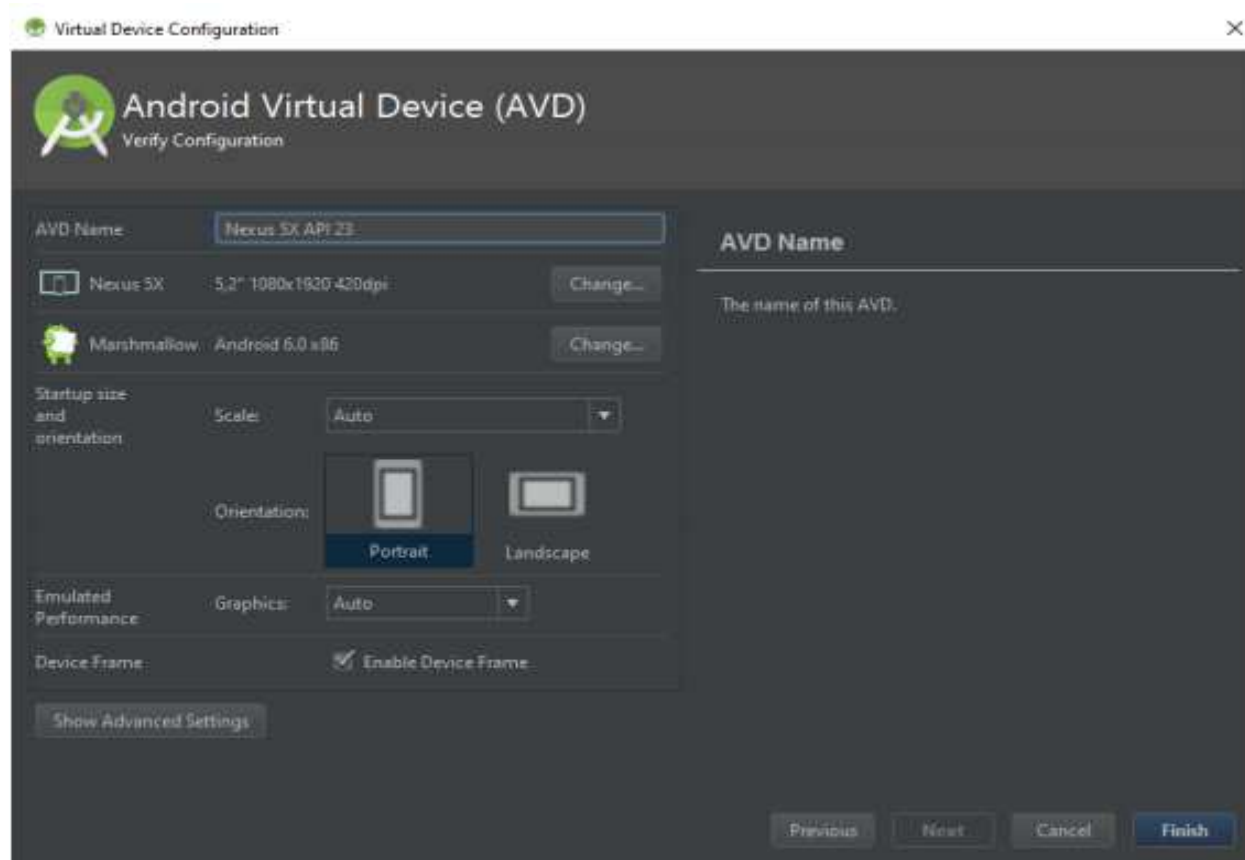


Рисунок 2.13 – Диалоговое окно создания нового AVD

В этом окне задаю нужную конфигурацию для создаваемого эмулятора устройства:

- AVD Name – имя создаваемого устройства;
- версия Android SDK, поддерживаемая эмулятором Android-устройства. Устройство имеет обратную совместимость со старыми версиями SDK, т. е. если выбрана версия Android 4.3, эмулятор будет поддерживать эту версию SDK и все, что находятся ниже версии SDK 4.3;
- Orientation – устанавливает ориентацию экрана по выбору, либо Portrait, либо Landscape;
- Graphics – устанавливается выбор того, как среда разработки Android Studio будет обрабатывать графику Hardware, которая использует графическую карту компьютера для ускорения рендеринга, Software, эмулирующую графику в программное обеспечение, использовав это, чтобы обойти проблемы с видеокартой компьютера, Auto, которая дает возможность эмулятору выбрать наилучший вариант, исходя из знаний о видеокарте компьютера.

После окончания настроек создания AVD менеджер создаст новое виртуальное устройство, название и версия API которого появятся в списке Yout Virtual Devices.

В зависимости от поддерживаемой версии API внешний вид виртуального устройства будет отличаться.

Окно эмулятора оформлено в виде телефона с дополнительной клавиатурой. На рисунке 2.14 показан внешний вид эмулятора. Эмулятор имитирует сенсорный экран реального мобильного устройства – в эмуляторе на экран нажимают левой кнопкой мыши.

В эмуляторе три виртуальных рабочих стола, перемещение по которым осуществляется с помощью кнопок со стрелками на навигационной панели устройства или передвижением курсора при нажатой левой кнопке мыши (в реальном устройстве – перемещая палец по экрану). Кроме ярлыков программы на рабочем столе можно размещать виджеты.



Рисунок 2.14 – Внешний вид AVD версии 4.3

Эмулятор, тем не менее, не поддерживает некоторые функциональности, доступные на реальных устройствах:

- входящие и исходящие сообщения. Однако можно моделировать обращения по телефону через интерфейс эмулятора;
- соединение через USB;
- видеочамера (однако есть имитатор работы видеочамеры);

- подключение наушников;
- определение статуса соединения;
- определение уровня заряда аккумуляторной батареи;
- определение вставки или изъятия карты памяти;
- соединение по Bluetooth;
- сканирование отпечатка пальца (в случае, если присутствует сканер).

Конечно, реальные телефоны несколько отличаются от эмулятора, но в целом AVD разработан очень качественно и близок по функциональности к реальному устройству.

2.3 Компоненты приложений в Android

Каждое Android-приложение запускается в своем собственном процессе и под своим собственным `userId`, который автоматически генерируется Android-ом во время развертывания. Поэтому приложение изолировано от других запущенных приложений, и не правильно работающее приложение не может беспрепятственно навредить другим Android-приложениям.

Основные компоненты Android:

- Activity;
- Views;
- Services;
- Contents Provider;
- Intents;
- Broadcast Receiver.

2.3.1 Activities

Активность - внешний пользовательский интерфейс для операции, которую может выполнить пользователь. Если упрощать, это просто текущий экран в качестве какой-то деятельности блока, своего рода рамки с одним действием пользователя. Например, активность может предоставить список пунктов меню, которые пользователь может выбирать или отображать фотографии со своими подписями. Или другой пример - приложение для обмена мгновенными сообщениями может использовать одно `activity` для отображения списка контактов, вторую `activity` - для того чтобы создать сообщение выбранному контакту, а третья - для просмотра истории сообщений или запустить настройки, и так далее.

Все `activity` текущего приложения работают вместе и образуют единый пользовательский интерфейс, но в то же время они являются независимыми друг от друга. Каждый из них реализуется как подкласс базового класса `activity`, обеспечивает создание окна, в котором программист может поместить визуальный интерфейс.

Приложение может состоять только из одного или нескольких из `activity`, как уже упоминалось ранее в качестве примера мессенджера. Какими именно будут `activity` и сколько их будет зависит от конкретного приложения и его структуры. Как правило, один из `activity` отмечен как первый, что

означает, что он будет предоставлен пользователю при запуске приложения. Однако одно activity может вызывать другое. Таким образом, переход от одного вида activity к другому осуществляется, когда текущая activity вызывает следующее.

Каждый вид деятельности предоставляет окно по умолчанию. Обычно окно создается в полноэкранном режиме, но оно также может занимать весь экран и оставаться наверху других окон. Activity также может использовать дополнительные окна - например, диалоговое всплывающее окно для взаимодействия с пользователем во время трудовой деятельности activity, или окно для того, чтобы предоставить соответствующую информацию при выборе каких-либо важных опций.

Визуальное содержимое окна строится с помощью иерархии визуальных компонентов (или представлений) - объекты, полученные из базового класса Views. Каждый компонент представляет собой пространство внутри прямоугольного окна. Родительские компоненты включают в себя дочерние и организуют их расположение. Иерархия компонентов может быть представлена в виде дерева, а также элементы, которые находятся в нижней части («листья»), и имеют ни один дочерний компонент. Таким образом, происходит интерактивное взаимодействие с пользователем. Например, может быть отображена на экране маленькая иконка и инициировано какое-либо действие, когда пользователь нажимает на нее. Операционная система Android имеет набор готовых визуальных компонентов, которые доступны для использования разработчиками. Набор включает в себя кнопки, текстовые поля, полосы прокрутки, меню, флажки, переключатели и многое другое.

Для того чтобы поставить эту иерархию в окне, необходимо вызвать метод `Activity.setContentView()`. Параметр метода является экземпляром Views, который лежит в корне иерархии.

2.3.2 Services

Services (сервисы) представляют собой компоненты, которые работают в фоновом режиме. Сервис, как правило, требует длительных операций для работы или удаленных процессов, но в целом это просто режим, который работает, когда приложение не находится в фокусе. Примером такого процесса может быть прослушивание музыки в то время как пользователь делает что-то другое или принимает данные по сети без текущей активности блокирования. Сервис сам по себе не обеспечивает пользовательский интерфейс, который не взаимодействует с пользователем, он запускается, работает и соединен с другими компонентами, например, activity. Он также может работать с системой. Сервисы реализованы в виде подкласса Service

2.3.3 Contents providers

Этот компонент управляет наборами данных, которые доступны для других приложений. Эти данные могут быть сохранены в файловой системе, в базе данных SQLite, в сети или в любом другом определенном месте, к

которому приложение может получить доступ. Contents provider через другое приложение может запрашивать данные и, если поставить соответствующие разрешения, он сможет изменять их. Например, Android-система содержит contents provider, который управляет контактной информацией пользователя. Он позволяет любому приложению, которое имеет соответствующие права вызвать составляющие этого компонента для чтения, записи или изменить информацию о конкретном человеке.

В более общем плане, contents provider может быть использован для чтения и записи данных, используемые приложением и не являются открытыми для других. Например, приложение Note Pad использует этот компонент для хранения записей.

Эти компоненты реализуются в качестве подкласса ContentProvider. И, чтобы гарантировать, что другие приложения могут выполнять операции над данными, необходимо обеспечить стандартный набор API.

2.3.4 Broadcast receivers

Этот компонент отвечает за распределение всей общесистемных сообщений, отслеживания и реагирования на действия. Многие сигналы приходят из системы, например, сообщает, что заряд батареи низкий, или экран выключен. Приложения могут также инициировать такое уведомление, например, чтобы сигнализировать о том, что информация загружается в устройство и доступна для использования. Как и сервисы, broadcast receiver не предоставляет пользовательского интерфейса, однако, он может создать уведомление в строке состояния, чтобы предупредить пользователя о том, что произошло событие. Тем не менее, чаще broadcast receiver взаимодействует с другими компонентами, чтобы выполнять самое минимальное количество работы. Тем самым он может инициировать сервисы, для выполнения действий, которые связаны с каким-то событием.

Компонент реализуется как подкласс BroadcastReceiver и каждая передача представляется как объект класса Intents.

2.3.5 Активация компонентов

Заострю внимание на уникальном свойстве системы Android, которое заключается в том, что любое приложение может использовать компоненты другого приложения. Например, если я хочу, чтобы пользователь в вашем приложении сделал фото с помощью камеры устройства, то возможно, есть другое приложение, которое делает это, что и мое приложение может использовать его функционал, вместо того, чтобы самому разрабатывать эту функцию в своем приложении. Вместо этого, я могу просто запустить компонент в приложение камеры, который делает снимок. Этот компонент сделает фото и вернет его в мое приложение, так что я могу использовать это фото в своем приложении. А для пользователя это будет казаться, как если бы функционал камеры на самом деле является частью моего приложения.

Когда система запускает компонент приложения, то она запускает процесс данного приложения (если он еще не запущен) и создает классы, необходимые для этого компонента. Например, если мое приложение запускает Activity приложения камеры, чтобы сделать снимок, то эта Activity, запускается в процессе, который принадлежит приложению камеры, а не процессу моего приложения. Поэтому, в отличие от приложений на большинстве других систем, приложения для Android не имеют единой точки входа (нет функции main (), например).

Поскольку система запускает каждое приложение в отдельном процессе с правами доступа к файлам, которые ограничивают доступ к этим файлам другим приложениям, приложения не могут непосредственно активировать компонент другого приложения. Однако система Android может. Поэтому, чтобы использовать компонент другого приложения, я должен сообщить об этом системе Android, что у меня есть намерение (Intents) запустить компонент какого-либо приложения, и система запустит этот компонент для меня.

Три из четырех типов компонент – Activity, Services и Broadcast Receivers активируются через асинхронные сообщения, называемые Intents 34 (намерение). Интенды (Intentss) связывают различные компоненты друг с другом в реальном времени (об интендах можно думать, как о посланниках, которые запрашивают какое-либо действие от компонента) и это безотносительно того, является ли данный компонент частью вашего приложения или другого.

Четвёртый тип компонентов – Contents Provider не активируется интендами (Intentss). Contents Provider активируется запросом от ContentsResolver [10].

Вывод

Во второй главе дипломной работы я рассмотрел архитектуру операционной системы Android, ее историю, преимущества. Проанализировал инструменты для разработки и отладки приложений Java Development Kit, Adnroid SDK, Adnroid Studio, провел их пошаговую установку и настройку под собственные предпочтения. Ознакомился с процессом работы в данных иснтрументах для разработки приложений и основными компонентами приложений на Android. Сравнил их между собой и оставил свой голос за Android Studio. Достоинства Android Studio:

- разработчик Google inc - та же корпорация, которая выпускает платформу, под которую я собираюсь писать приложение;
- встроенный SDK;
- очень удобный конструктор интерфейсов;
- доступная структура проекта;
- удобный дизайн;
- удобное отслеживание логов.

В целом главным достоинством студии для меня стала тесная интеграция с Android SDK, что напрямую улучшает не только удобство разработки, но и качество приложений.

Операционная система Android является связующим звеном между собственно аппаратом и его программным обеспечением, а также позволяет устанавливать на устройство нужные дополнительные приложения от других разработчиков - мультимедийные, офисные, коммуникационные. Для ОС Android созданы около полутора миллиона прикладных программ, существенно расширяющих и улучшающих функциональность смартфона, делающих его уникальным и максимально полезным для владельца.

Android - это программная платформа для мобильных устройств, которая включает в себя операционную систему, программное обеспечение, промежуточное программное обеспечение (middleware), а также основные пользовательские приложения (электронная почта-клиент, карта, браузера, календарь, контакты и т.д.).

3 Практическая реализация

Я создаю Мессенджер-приложение, которое работает по Bluetooth и беспроводной сети по защите данных на языке Java в среде разработки Android Studio для операционной системы Android и разрабатываю программный продукт для ОС Windows, который шифрует и дешифрует текст, реализуя таким образом криптографическую систему защиты данных на двух платформах.

3.1 Структура Android-приложения

На рисунке 3.1 приведено структурное содержание проекта. Проект состоит из папок и файлов.

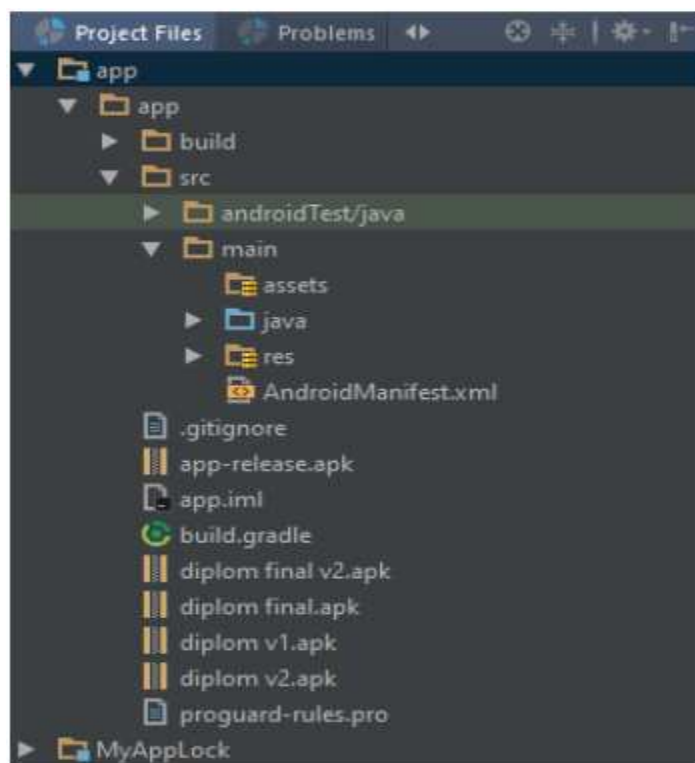


Рисунок 3.1 – Структура проекта

Основные содержимые структуры проекта:

- папка `src` хранит весь исходный код на Java. В этой папке находится основной файл для работы. Здесь же будут находиться новые классы;
- папка `java` хранит код приложения;
- папка `res` содержит ресурсы (`res` – resources), необходимые приложению (текстовые строки, различные графические элементы: иконки, картинки, анимации и пр.). Некоторые из подкаталогов генерируются Android Studio при создании проекта, другие необходимо добавлять самостоятельно, используя predetermined имена. Обычно в ресурсы включают следующие подкаталоги (рисунок 3.2):

- папки `res/mipmap-hdpi/`, `res/ mipmap -ldpi/`, `res/ mipmap -mdpi/`, `res/mipmap -xhdpi/`, `res/ mipmap -xxhdpi/` для хранения значков используемых в приложении. Значки в каждой папке рассчитаны на соответствующее разрешение экрана мобильного устройства;
- `res/layout/` папка для XML-файлов компоновки (компоновка графических элементов управления для окон приложения);
- `res/menu/` папка для XML-файлов меню;
- `res/values/` папка для строковых ресурсов, массивов и т. д.;
- `AndroidManifest` – один из самых важных файлов в структуре Android-проекта. В нем декларируются все компоненты приложения, его имя, разрешения и пр.

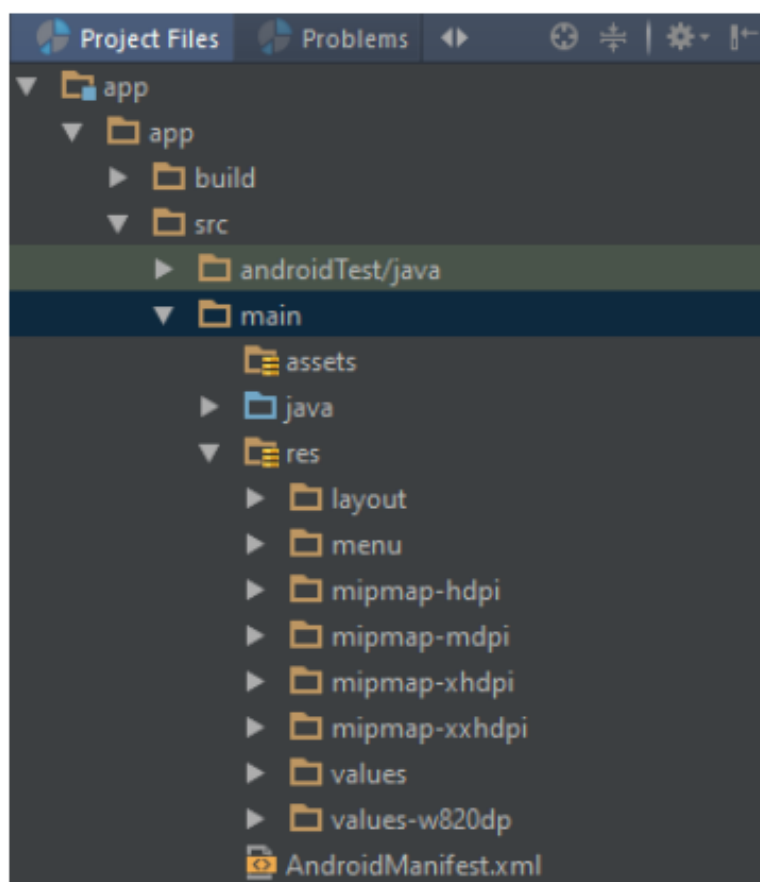


Рисунок 3.2 – Структура каталога ресурсов

Здесь следует отметить некоторые ограничения относительно создания папок файлов ресурсов проекта. Android поддерживает только линейный список файлов в пределах предопределенных папок под каталогом `res/`. Например, он не поддерживает вложенные папки под папкой для XML-файлов компоновки (или другими папками в каталоге `res/`).

3.2 Алгоритм работы Android-приложения

На рисунке 3.3 показана блок-схема работы меню приложения системы защиты данных, где П1 – это пользователь 1, а П2 – пользователь 2. При открытии приложения на первом окне будет изображено окно регистрации/логина.

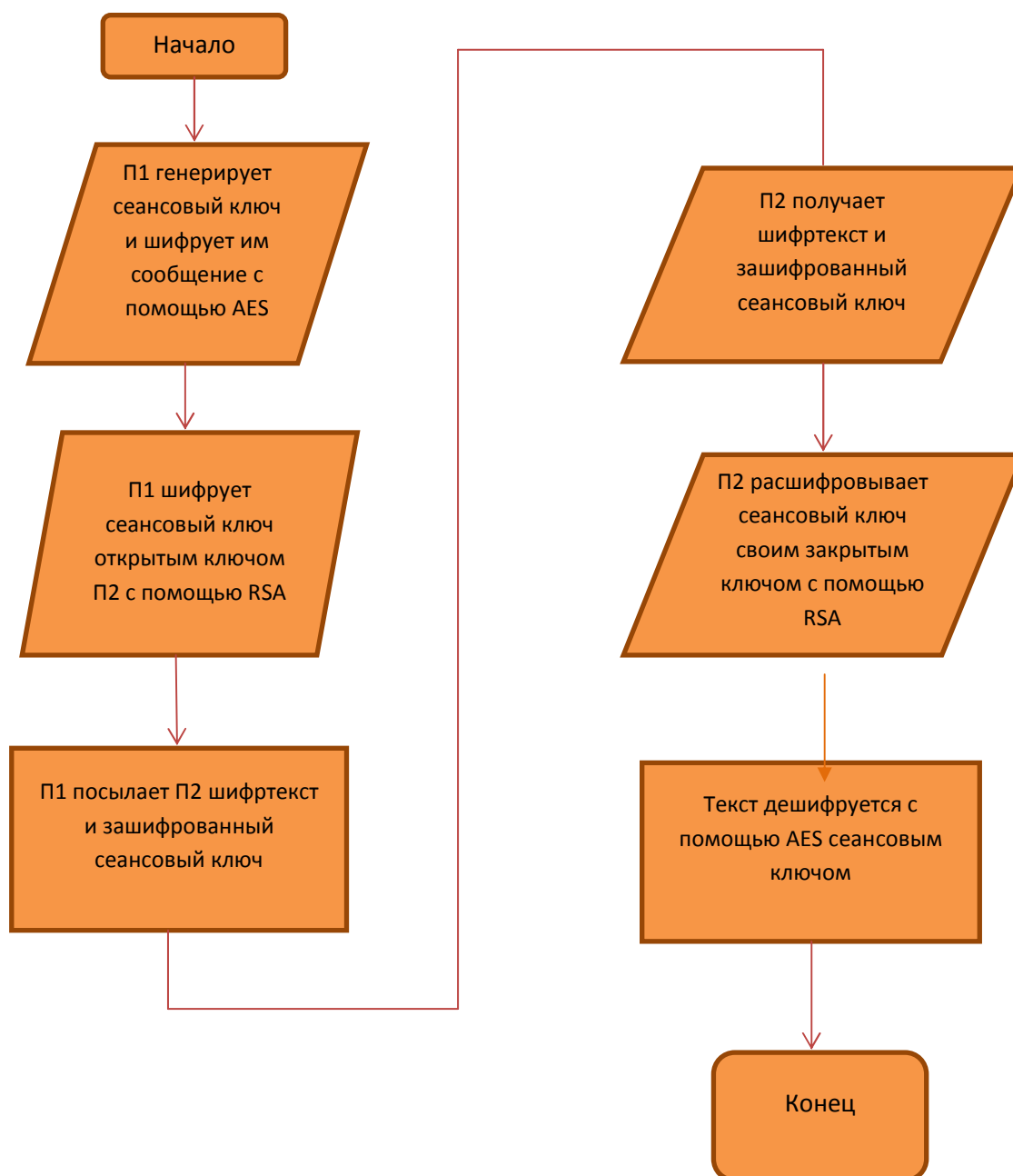


Рисунок 3.3 – Блок-схема меню приложения

Приложение использует два алгоритма шифрования: AES и RSA. Для работы с приложением сначала регистрируется два пользователя, указав свой логин и пароль, соответственно. После регистрации пользователи выбирают каким способом производить чат между собой. Есть два способа:

- 1) по Bluetooth
- 2) по беспроводной сети

Алгоритм шифрования сообщений следующий:

1. Пользователь 1 генерирует сеансовый ключ и шифрует им сообщение с помощью алгоритма AES;
2. Пользователь 1 шифрует сеансовый ключ открытым ключом Пользователя 2 с помощью алгоритма RSA;
3. Пользователь 1 посылает зашифрованное сообщение и зашифрованный сеансовый ключ Пользователю 2;
4. Пользователь 2 получает зашифрованное сообщение и зашифрованный сеансовый ключ;
5. Пользователь 2 расшифровывает сеансовый ключ своим закрытым ключом с помощью RSA;
6. Пользователь 2 дешифрует сообщение сеансовым ключом.

При работе с приложением можно выбрать несколько настроек соединения:

1. подключение к устройству – безопасно;
2. подключение к устройству – небезопасно;
3. сделать видимым.

3.3 Компиляция и установка Adnroid-приложения

Теперь запускаю приложение на виртуальном устройстве. Для этого в AVD Manager добавляю устройство (рисунок 3.4). В настройках устройства указываю разрешение экрана, версию Android и размер выделения памяти на диске. Листинг приложения представлен в приложении А.

После того, как добавил устройство, запускаю эмулятор (рисунок 3.5).

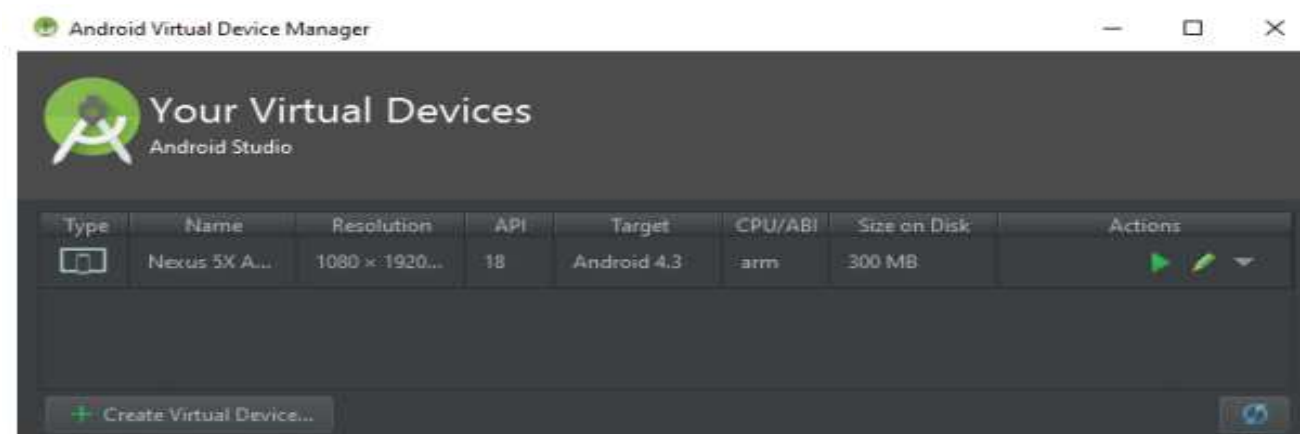


Рисунок 3.4 – Окно AVD Manager



Рисунок 3.5 – Виртуальное устройство Android

После этого при нажатии на кнопку запуска приложения в ToolBar (рисунок 3.6), приложение загрузится и запустится на подключенном виртуальном устройстве.



Рисунок 3.6 – ToolBar в Android Studio

Описание процесса запуска отладки под Windows 10 в Android Studio, по шагам:

- 1) Перед началом тестирования приложения нужно разрешить режим отладки USB. Для этого необходимо открыть Настройки (рисунок 3.8) → Функции для разработчиков → Подключить режим «Отладка по USB» (рисунок 3.9). Высветится предупреждение, подтвердить положительно - «Да».

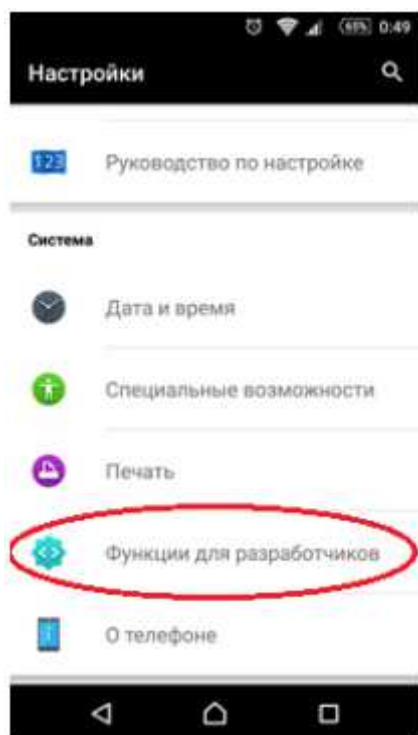


Рисунок 3.8 – Настройка функций разработчика

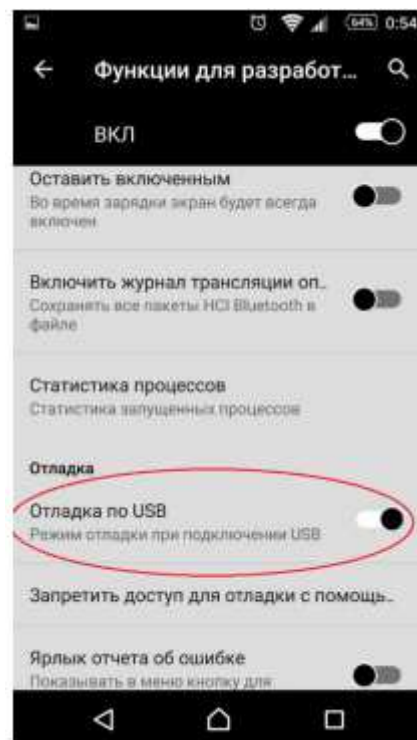


Рисунок 3.9 – Настройка отладки по USB

2) Подключение мобильного устройства с ОС Android через USB к компьютеру. На смартфоне должен определиться режим «Подключен как камера (PTP)». Компьютер обнаружит новое устройство, и затем запустится мастер установки драйвера.

3) Далее запускаю Android Studio. Нажимаю Tools → Android и ставлю галочку напротив строки «Enable ADB Integration» (ADB - Android Debug Bridge). После этого нужно настроить Android Studio так, чтобы при нажатии на зеленую кнопку «Run» приложение сразу устанавливалось и запускалось на подключенном смартфоне. Нажимаю Run → Edit Configurations.

После этого драйвер определяет устройство (рисунок 3.10), и при нажатии на выбранное устройство приложение установится с логотипом Алматинского университета энергетики и связи (рисунок 3.11), запустится на подключенном устройстве и попросит выбрать способ подключения (рисунок 3.12).

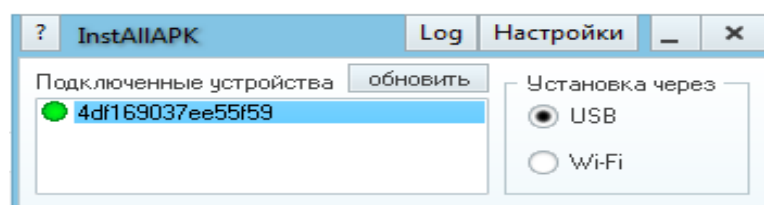


Рисунок 3.10 – ToolBar в Android Studio

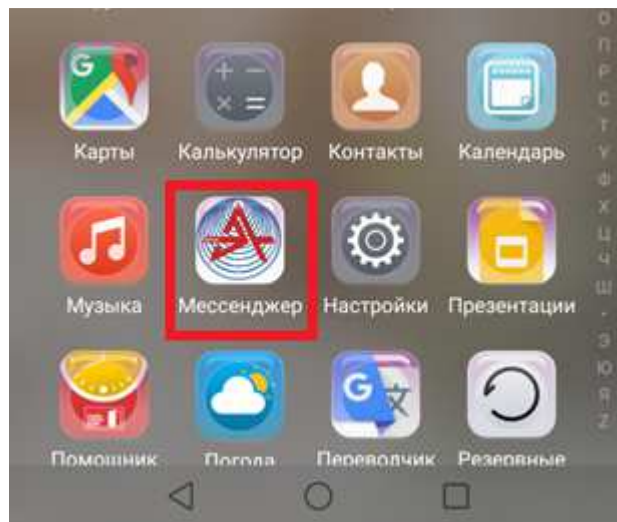


Рисунок 3.11 – Установленное приложение мессенджер

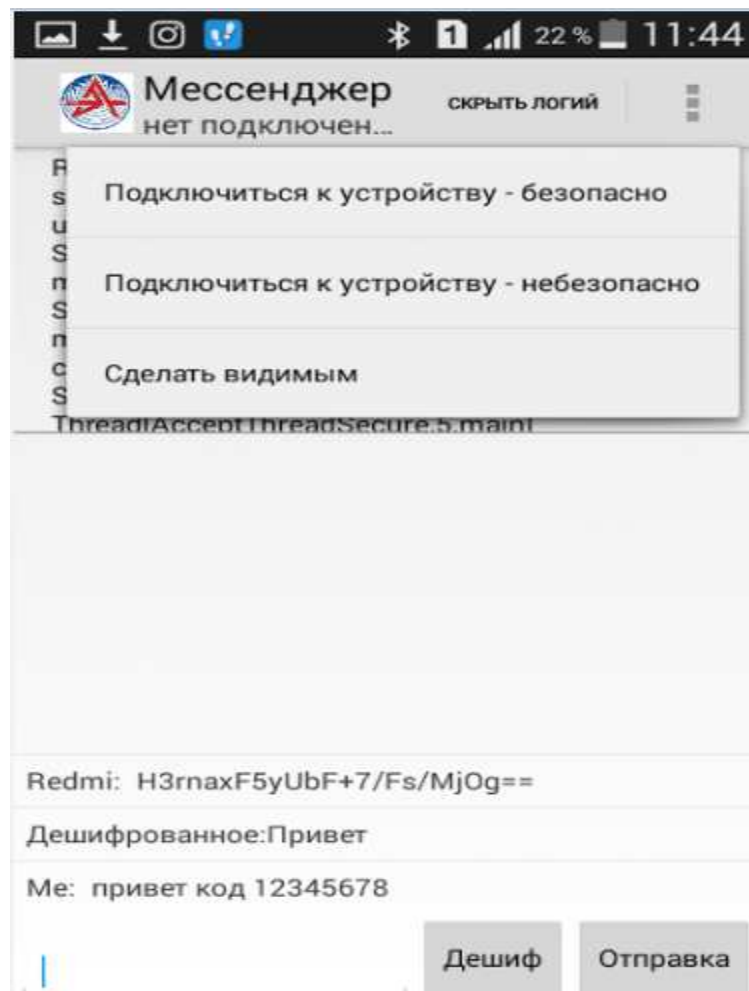


Рисунок 3.12 – Подключение абонента

3.4 Тестирование Android-приложения

После запуска приложения на устройстве откроется окно с пунктами меню, приведенное на рисунке 3.13. Из меню надо выбрать один пункт для дальнейшей работы

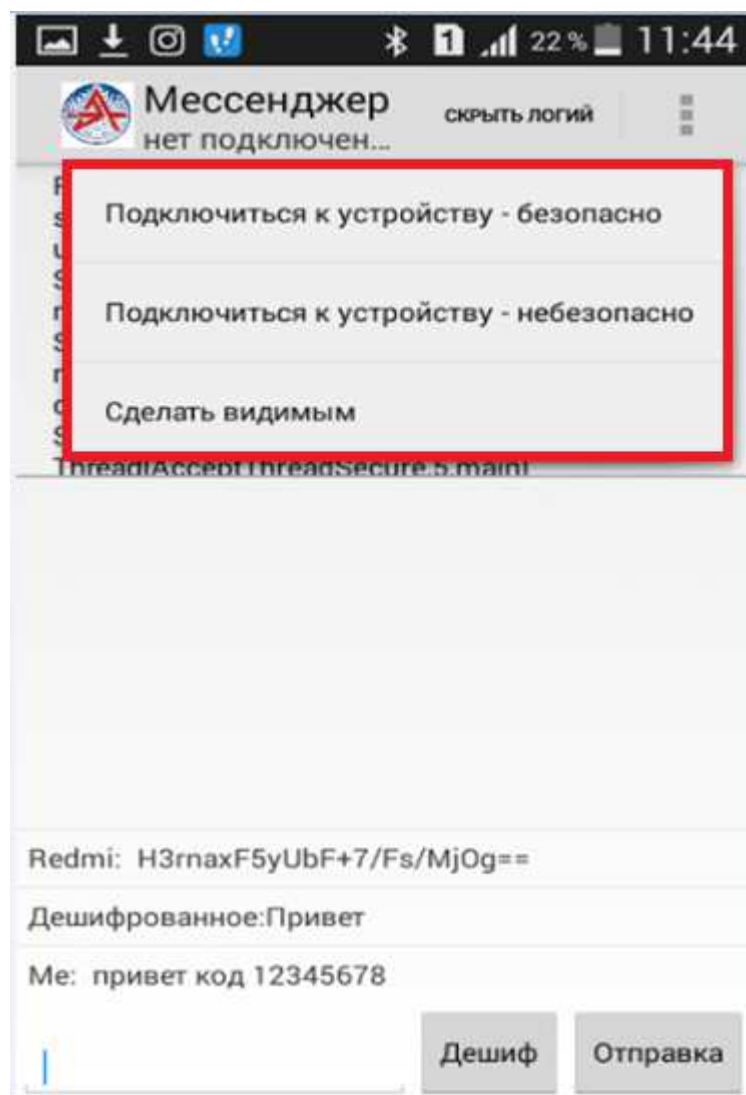


Рисунок 3.13 – Список меню соединений

Дальше откроется диалоговое окно с сообщениями. Первый пользователь отправляет слово “Привет” (рисунок 3.14). Далее программа мигом зашифровывает слово с помощью алгоритма шифрования. Зашифрованное слово отправляется второму пользователю. Второй пользователь получает зашифрованное слово (рисунок 3.15). После этого программа дешифрует зашифрованное слово, а значит представляет текст пользователю в понятном для чтения виде.

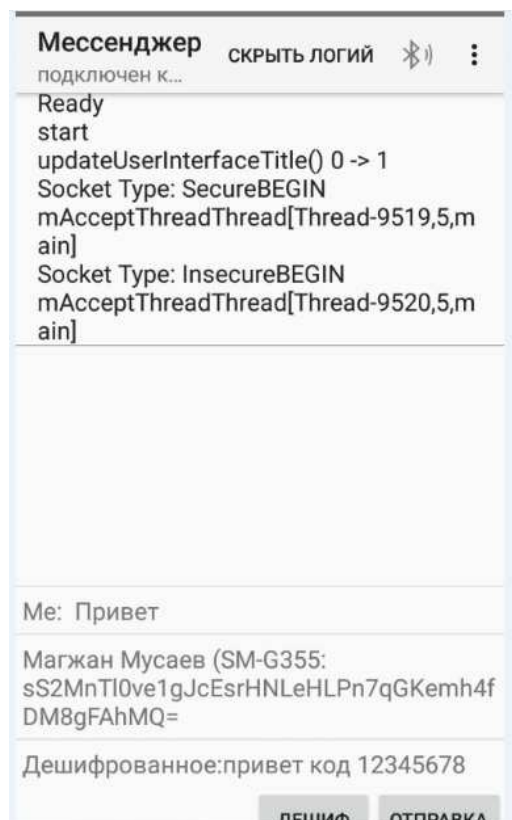


Рисунок 3.14 – Пользователь 1



Рисунок 3.15 – Пользователь 2

3.5 Структура программы для Windows

Создаю проект Visual Studio.

Создаю новое визуальное решение студии, содержащее проект Visual C# 'WPF Application', называя его 'WPFChatExample'.

Щелкаю правой кнопкой мыши на только что созданный проект и выбираю «Свойства». Надо проверить, что «Target Framework» - «.NET Framework 4», а не «Профиль платформы .NET Framework 4». После этого получается что-то похожее на это (рисунок 3.16).

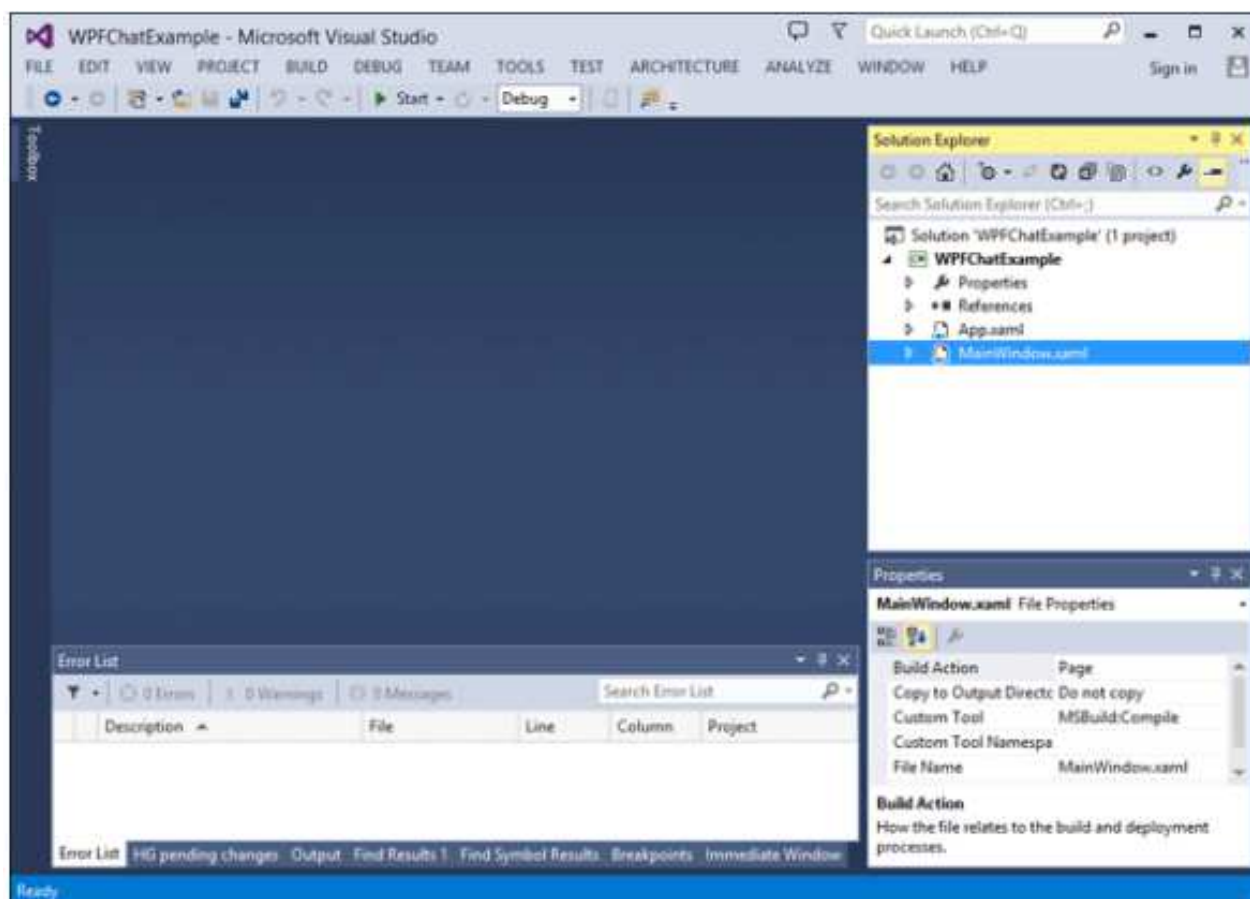


Рисунок 3.16 - Созданная визуальная студийная программа под названием 'WPFChatExample'

Пакет загрузки NetworkComms.Net содержит библиотеки DLL для всех поддерживаемых платформ, но меня интересует только Net40> Release> Complete DLL. Скопируем эту DLL в то же место, что и решение, которое я создал на шаге 1.

Теперь мне нужно добавить ссылку на проект только что добавленной DLL-библиотеки NetworkComms .Net. Щелкаю правой кнопкой мыши по проекту WPFChatExample и выберите «Добавить ссылку ...». В открывшемся окне выбираю вкладку «Обзор» и DLL, которую я только что добавил.

Если разворачивать папку «Ссылки» в проекте, нахожу ссылку NetworkComms .Net, которую я только что добавил (рисунок 3.17):

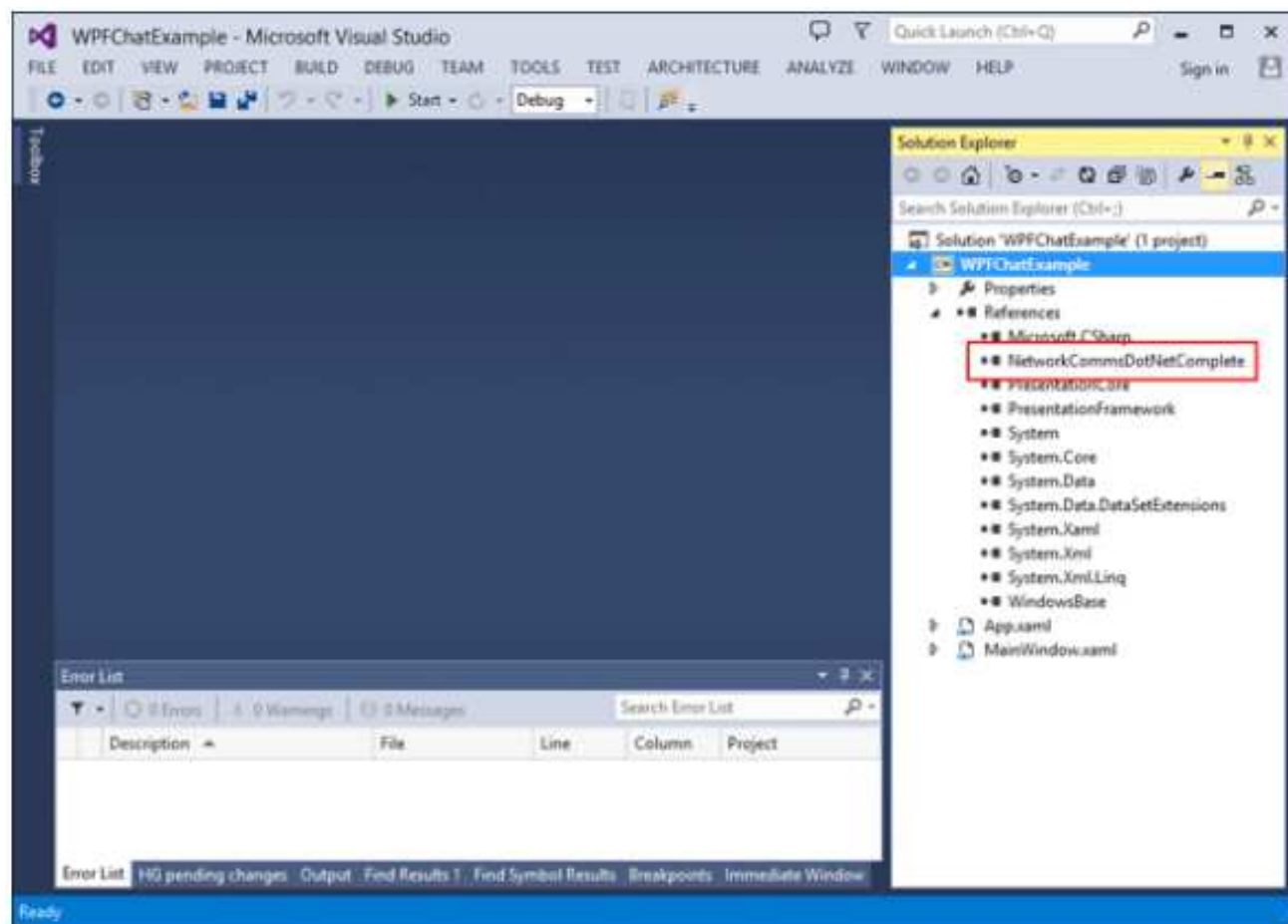


Рисунок 3.17 - Приложение WPFChatExample WPF, содержащее ссылку на полную DLL-сеть NetworkComms .Net.

Следующий шаг - создать класс-оболочку для сообщений, которые я буду отправлять и получать, то есть один объект, который я отправляю и получаю, который содержит всю необходимую информацию. Щелкаю правой кнопкой мыши проект и выбираю «Добавить»> «Создать элемент ...». Это должно вызвать окно «Добавить новый элемент», список параметров, которые я могу добавить в проект. Убеждаюсь, что выбран пункт «Класс», а в нижней части окна введите имя «ChatMessage.cs». Теперь нажимаю «Добавить». Новый файл класса должен открыться автоматически, и у нас должна получиться форма.

Далее я собираюсь добавить некоторые переменные класса, чтобы помочь отслеживать текущее состояние приложения. Я хочу отслеживать:

- 1) последние сообщения, которые я получил;
- 2) максимальное количество раз, когда я буду передавать сообщение;
- 3) необязательный ключ шифрования;
- 4) локальный индекс, который я буду использовать при отправке новых

сообщений.

Чтобы отслеживать эти элементы нужно добавить соответствующий код в начале класса.

Далее следует пять методов, которые будут привязаны к элементам в макете WPF. Они будут использоваться для отправки наших сообщений, переключения шифрования, переключения режима локального сервера и правильного закрытия всех окон после того, как я закончу работу с приложением. Листинг программы представлен в приложении Б.

3.6 Установка и тестирование программы для Windows

После отладки приложения я распаковываю его на рабочий стол в папку вместе со всеми подключенными библиотеками, на которые ссылается приложение (рисунок 3.18).

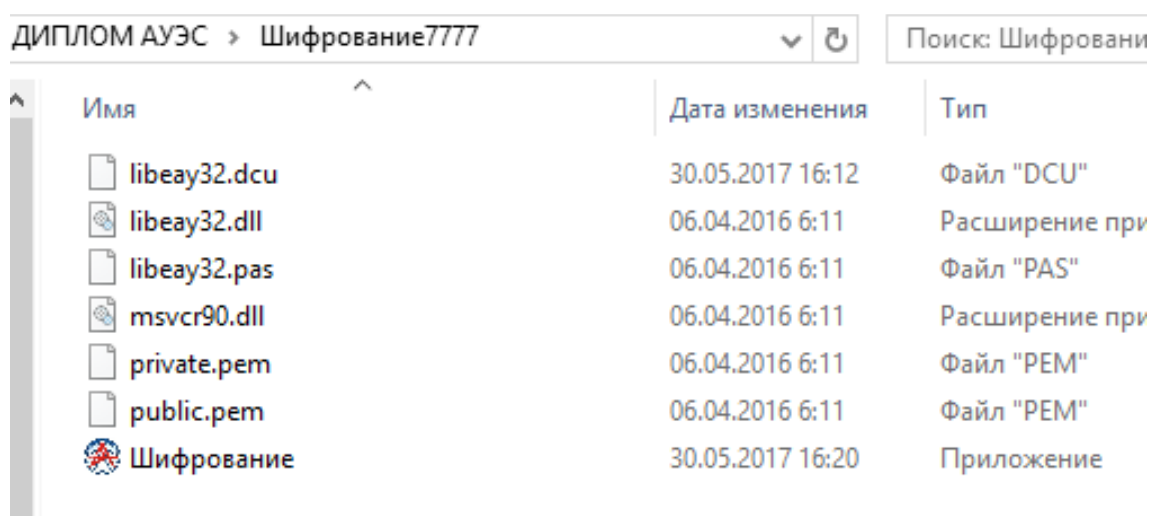


Рисунок 3.18 – Содержимое папки с программой

Мой программный продукт называется «Шифрование» и его иконкой является логотип Алматинского университета энергетики и связи (рисунок 3.19).



Рисунок 3.19 – Иконка программного продукта

Далее запускаю exe-файл «Шифрование» и наблюдаю окно, в котором в правом верхнем углу расположены вкладки «Файл» и «Автор», чуть ниже расположены 4 кнопки: «шифрование», «дешифрование», «очистить», «выйти». Также в программе содержится 4 окна: «данные для шифрования», «шифрованные данные», «данные для дешифрования», «дешифрованные данные», «лог операций». При запуске программы в первой окно «Данные для шифрования» помещаю текст «Алматинский университет энергетики и связи», который можно заменить любым текстом, остальные окна пустуют (рисунок 3.20). Функционал каждого элемента из числа перечисленных выше будет описан ниже по-отдельности.

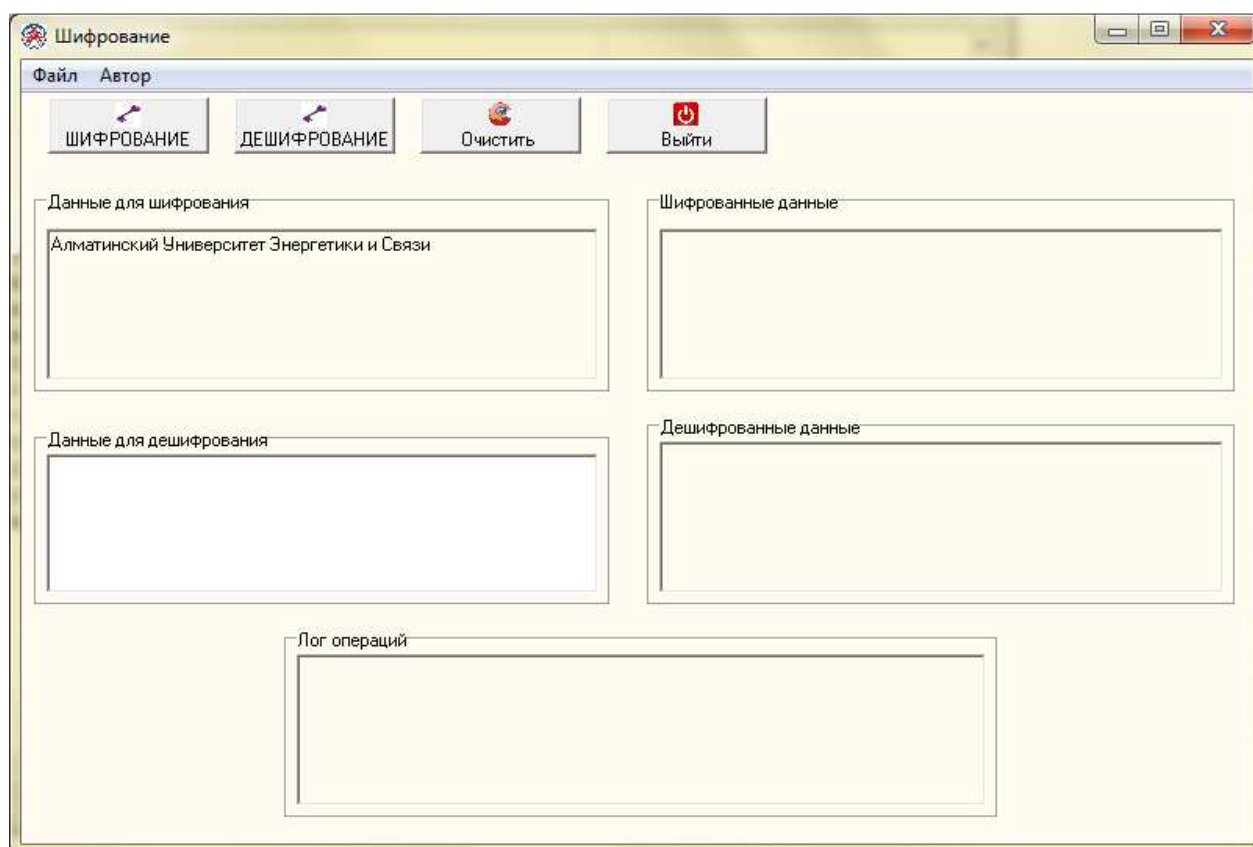


Рисунок 3.20 – Интерфейс программы при запуске

Вкладка «Файл» содержит в себе 5 элементов (рисунок 3.21):

- инструкция, которая выводит информацию об алгоритме шифрования данных (рисунок 3.22);
- SHA1 – алгоритм криптографического хеширования версии 1;
- SHA256 – однонаправленная хеш-функция семейства SHA2 версии;
- SHA512 – однонаправленная хеш-функция семейства SHA2 версии;
- «выйти» – функция выхода из программы.

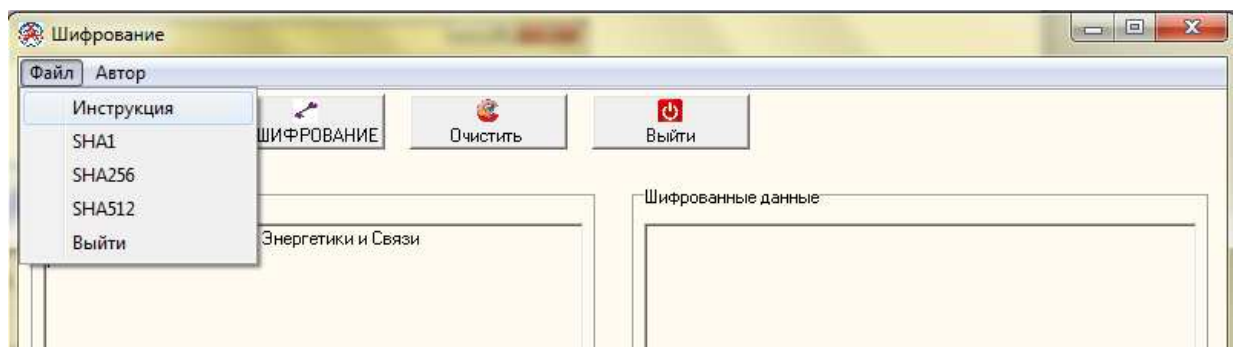


Рисунок 3.21 – Содержимое вкладки «Файл»

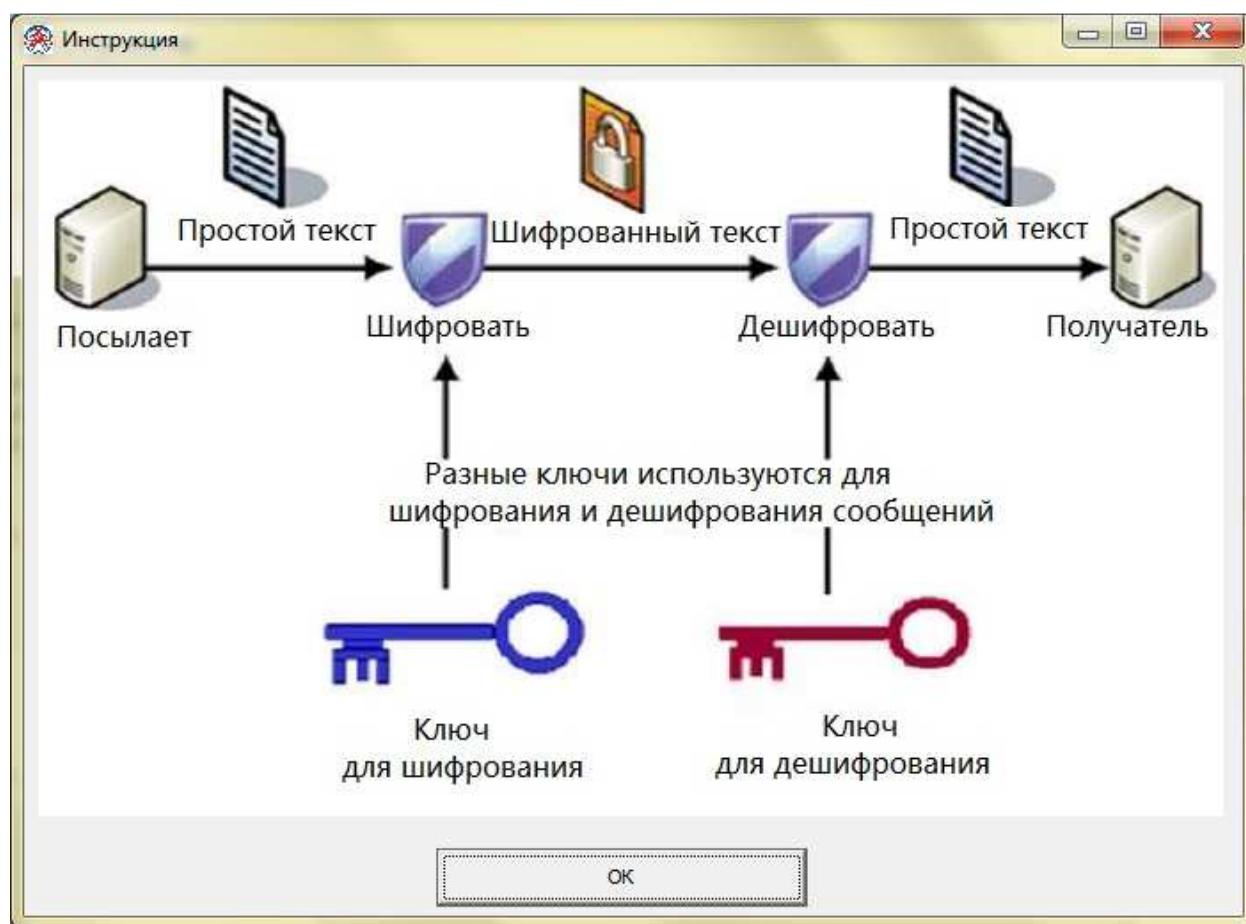


Рисунок 3.22 – Инструкция программы

Следующая вкладка «Автор», в которой дана информация об авторе программного продукта (рисунок 3.23).

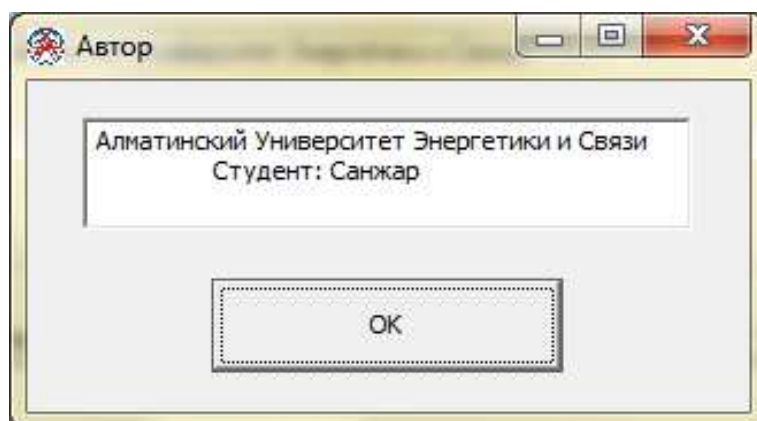


Рисунок 3.23 – Вкладка Автор

Кнопка «Шифрование» шифрует данные, которые вводятся в поле «Данные для шифрования» и выводит зашифрованный текст в поле «Шифрованные данные» (рисунок 3.24).

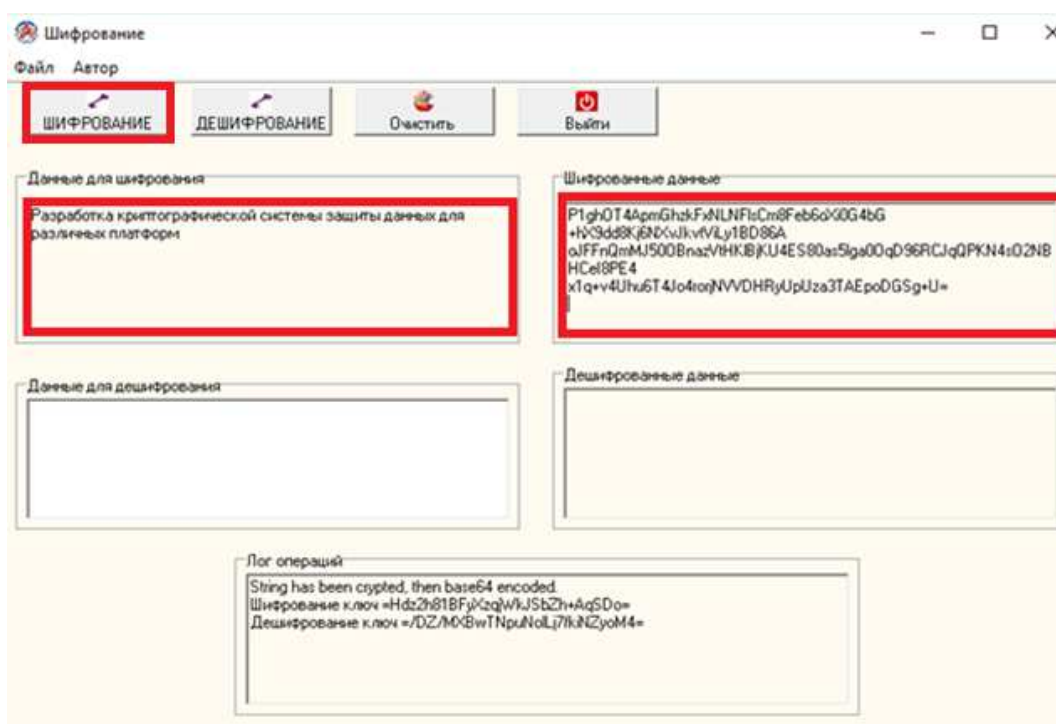


Рисунок 3.24 – Шифрование данных

По нажатию кнопки «Дешифрование» зашифрованный текст в поле «Данные для дешифрования» преобразуется в открытый читабельный текст, который выводится в поле «Дешифрованные данные». Перед нажатием кнопки «Дешифрование» необходимо поместить зашифрованный текст в соответствующее ему поле (рисунок 3.25).



Рисунок 3.25 – Дешифрование данных

В поле «Лог операций» выводится отчет последовательных действий пользователя. В данном поле отображаются ключи для шифрования и дешифрования, результаты проведенных операций (рисунок 3.26).

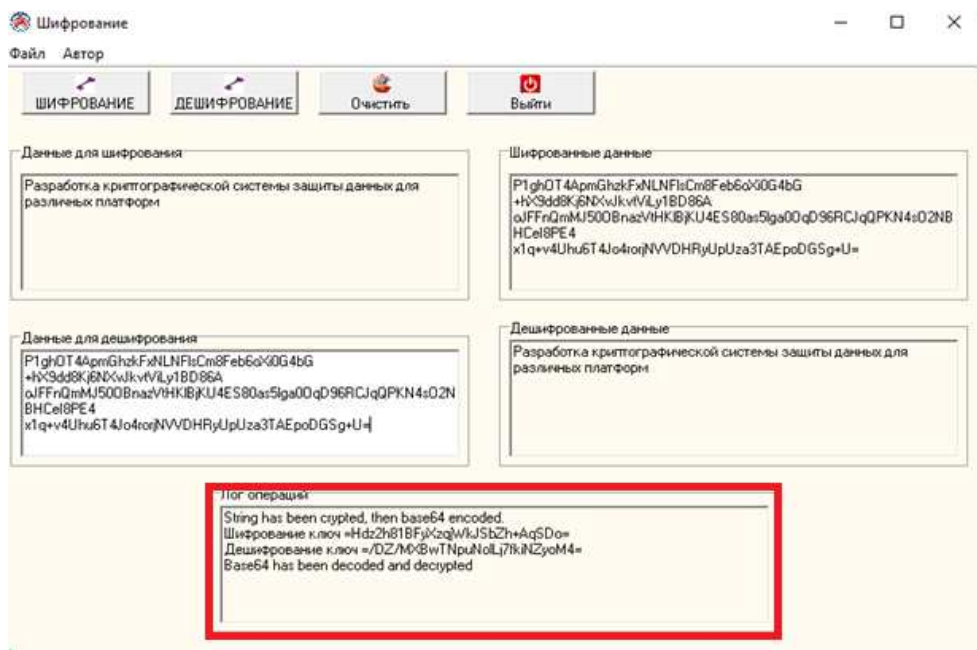


Рисунок 3.26 – Лог операций

По нажатию кнопки «Очистить» каждое поле очищается от текста и становится пустым (рисунок 3.27). По нажатию кнопки «Выйти» программа закрывается.

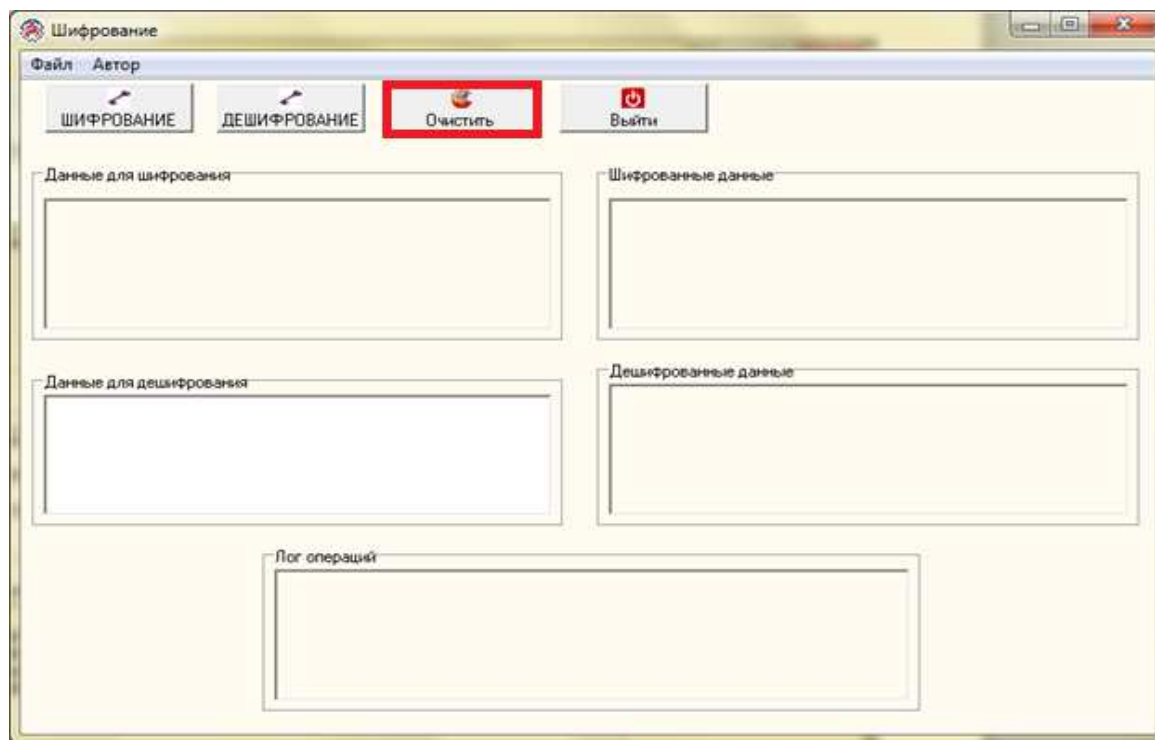


Рисунок 3.27 – Очистка полей

Созданная мною программа на ОС Windows работает и выполняет все свои функции корректно. Программа выполняет следующие функции:

- шифрует текст;
- дешифрует зашифрованный текст;
- выводит отчет о каждой операции.

Вывод

В данной главе я создал программное обеспечение для шифрования на Android и Windows, описал структуру проекта, структуру его каталогов и файлов ресурсов. Описал процесс подключения мобильного устройства к компьютеру через USB-кабель для тестирования приложения. Описал весь функционал приложения.

Безопасности мобильного устройства следует уделять особое внимание. Во-первых, смартфон гораздо легче потерять, чем скажем, ноутбук или планшет, во-вторых, смартфоны точно так же, как и персональные компьютеры (даже в большей мере) подвержены вирусным атакам. Помимо надежного антивирусного программного обеспечения на смартфонах обязательно должны быть установлены средства блокировки пользовательских данных.

Приложение «Мессенджер» на ОС Android позволяет пользователям осуществлять переписку, надежно защищая важную информацию пользователей от посторонних лиц путем шифрования информации гибридным алгоритмом шифрования. Приложение не допускает присоединения посторонних лиц к чату без авторизации. Приложение реализовано в среде разработки Android Studio.

Программа на ОС Windows также позволяет защитить важную информацию от посторонних лиц, шифруя и дешифруя ее тем же алгоритмом шифрования. Программа разработана в среде разработки Microsoft Visual Studio.

Таким образом, программные продукты имеют ряд преимуществ:

- безопасное соединение абонентов;
- надежная защита и передача информации;
- удобный и простой интерфейс;
- минимальные системные требования;
- высокая скорость работы;
- стирание всех данных после окончания сессии.

4 Технико-экономическое обоснование

4.1 Характеристика проекта

Целью данного дипломного проекта является разработка системы защиты данных на платформах Android и Windows, имеющих настраиваемый пользовательский интерфейс для запуска приложений на смартфоне, соответствующего всем современным методам разработки приложений под Android OS и на компьютере под Microsoft Windows. Разработка приложения производится в среде Android Studio версии 1.5.1 на языке Java и в Microsoft Visual Studio на языке C#.

Защита информации в местах хранения успешно реализуется с помощью шифрования и программ-блокировщиков. Данные всегда находятся на носителях в зашифрованном виде и становятся доступны только после ввода пароля.

Данный программный продукт выступает на ненасыщенном рынке, поэтому его потенциальная конкурентоспособность достаточно высокая. В случае если цена на такой или аналогичный продукт будет выше цены на аналогичный продукт известной фирмы, конкурентоспособность будет снижена, в случае специального занижения цены, производство продукта будет являться экономически невыгодным, что также может служить препятствием к его производству.

Данный проект был осуществлен с использованием группы специалистов, организованной по иерархическому принципу. В группу вошли: руководитель проекта и программист-разработчик. Руководитель проекта должен оказывать помощь в разработке подробного календарного графика выполнения работы и контроль его соблюдения, помощь в составлении плана выпускной квалификационной работы, проводит беседы и консультации, проверяет выполненную дипломную работу как по частям, так и в целом. Программист-разработчик должен подготовить теоретическое обоснование, создать проект, разработать алгоритм и интерфейсную идеологию, реализовать ПО. Таким образом, на программисте-разработчике лежит ответственность за планирование и общая ответственность за реализацию проекта. Программист-разработчик обязан реализовать программные модули системы, провести рабочее тестирование проекта. Технико-экономическое обоснование разработки и внедрения в моей работе содержит следующее:

- определение трудоемкости разработки мобильного приложения;
- расчет затрат на разработку мобильного приложения;
- определение возможной цены разработанного мобильного приложения;
- оценку социально-экономических результатов функционирования приложения.

4.2 Трудоемкость разработки ПП

Целью данной части дипломной работы является определение трудоемкости разработки ПП, построение линейного графика разработки ПП, оценка удельного веса творческого труда по этапам разработки ПП, себестоимости дипломной работы, определение прибыли и договорной цены ПП, оценка научной и научно-технической результативности работы. Трудоемкость работы определялась согласно нормам времени на проведение расчетов, анализа и исследований. Разобьем процесс разработки ПП на этапы по каждому виду работ и определим ожидаемую трудоёмкость выполнения каждого вида работ. На каждом этапе проведения дипломной работы по каждому виду работ определяется уровень квалификации исполнителей (таблица 4.1).

Таблица 4.1 – Распределение работ по этапам и видам и оценка их трудоемкости

Этапы разработки ПП	Вид работы на данном этапе	Трудоемкость разработки ПП, чел. х ч.
1 этап	Постановка задач	8
2 этап	Разработка и утверждение технического задания на разработку ПП	8
3 этап	Ознакомление с существующими методами разработки защиты	12
4 этап	Подбор и изучение литературы по теме: «Разработка ПП для защиты данных»	14
5 этап	Составление аналитического обзора состояния разработки темы разработки ПП	12
6 этап	Проведения анализа разных методов разработки комплексной защиты	12
7 этап	Оформление теоретической части темы дипломного проекта	12
8 этап	Разработка экспериментальной части дипломного проекта	12
9 этап	Выбор среды разработки	12
10 этап	Разработка математических расчетов и написание программного обеспечения	12
11 этап	Реализация проекта	12
12 этап	Отладка приложения	12
13 этап	Оформление отчета и составление выводов о проделанной работе	12
14 этап	Тест	12
15 этап	Итог разработки ПП	12
16 этап	Внедрение	8
17 этап	Тестирование	8
ИТОГО трудоемкость выполнения дипломной работы		190

Количество дней, затраченных на осуществление цели – 32 рабочих дня.

4.3 Расчет затрат на разработку ПП

Расчет себестоимости проводят по тем расходам, которые были сделаны при разработке ПП. Затраты на проведение дипломной работы относятся к предпроектным затратам – это одноразовые затраты на все работы, которые выполняют разработчики ПП.

Затраты определяются путем сложения статей калькуляции себестоимости: материалы, спецоборудование для научных и экспериментальных работ, заработная плата, начисление на фонд оплаты труда, затраты на работы, которые выполняются другими организациями; другие прямые затраты; накладные расходы [11].

Данный проект предусматривает разработку и тестирование программы, которая помогает опробовать на практике шифрование и дешифрование сообщения, а также дает возможность обезопасить установленные приложения на смартфоне на выбор. Следовательно, для достижения поставленных целей, необходимо подсчитать затраты на разработку и тестирование программы, стоимость итоговой программы, закупку соответствующего оборудования, а так же окупаемость затрат (таблица 4.2).

Таблица 4.2 – Затраты на материальные ресурсы

Наименование материального ресурса	Единица измерени я	Количество расходов из данного материала	Цена за единицу, тг	Итого, тг
Персональные компьютеры с предустановленной операционной системой MS Windows 7 и офисным пакетом MS Office	шт	1	120000	120000
Ноутбук Lenovo Z50-70	шт	1	160000	160000
USB Manhattan 307178, A(M)/microB(M), черный	шт	2	200	400
Смартфон Xperia Z3	шт	1	140000	140000
Android Studio	шт	1		
Java Development Kit	шт	1		
ИТОГО затраты на материальные ресурсы				420400

Согласно данной таблице материальные затраты на проект составят 420400 тенге. Все материальные затраты лягут на основные средства, как отдельно (компьютеры), так и комплектующие.

Расчет затрат на электроэнергию приведен в таблице 4.3.

Таблица 4.3 – Затраты на электроэнергию

Наименование оборудования	Кол-во	Пасп. мощн, кВт	Время работы оборуд., ч	Коэф. исп. мощности	Цена электро энергии, $\frac{тг}{кВт * ч}$	Итого
Компьютер	1	0,350	672	235,20	15,19	1411,20
Раб. станция	1	400	672	23,52	15,19	141,12
Освещение	1	0,400	148	59,20	15,19	355,20
Кондиционер	1	1,400	148	207,20	15,19	1243,20
ИТОГО						3156,48

Общая сумма затрат на электроэнергию (Эн) рассчитывается по следующей формуле:

$$\mathcal{E}_n = \sum_{i=1}^n M_i * K_i * T_i * C \quad (1.1)$$

где M_i - паспортная мощность i -го оборудования, кВт;

T_i - время работы i -го оборудования за год в рабочие дни, ч.;

K_i - коэффициент использования мощности i -го электрооборудования (обычно принимается $K_i=0,7..0,9$);

C - цена электроэнергии, $\frac{тг}{кВт*ч}$;

i – вид электрооборудования;

n – количество электрооборудования;

Расчет времени работы T i -го оборудования за год в рабочие дни рассчитывается по следующей формуле:

$$T_i = t_i * 275 \quad (1.2)$$

где t_i – время работы оборудования в день, 275 – количество рабочих дней в году.

$$T_i = 8 * 275 = 2200 \text{ ч,}$$

$$\mathcal{E}_n = 1411,20 + 141,12 + 355,20 + 1243,20 = 3156,48 \text{ тг.}$$

Для реализации поставленных целей необходим программист-тестировщик. Затраты на оплату труда тестировщику приведены в таблице 4.4.

Таблица 4.4 – Затраты на оплату труда

Категория сотрудника	Квалификация	Часовая ставка тг/ч	Заработная плата за год, тг
Руководитель	Старший преподаватель	2000	40000
Программист-разработчик	Начальная	1000	100000
ИТОГО			140000

Общая сумма затрат на оплату труда (Z_{mp}) определяется по формуле:

$$(Z_{mp}) = \sum ЧC_i * T_i, \quad (5.5)$$

где $ЧC_i$ - часовая ставка i-го работника, тг;

T_i - трудоемкость разработки модели, чел.×ч;

i - категория работника;

n - количество работников, занятых разработкой ПП.

Часовая ставка руководителя составляет 2000 (тг/ч), трудоемкость разработки – 20 ч.

Часовая ставка разработчика составляет 1000 (тг/ч), трудоемкость разработки – 100 ч. Рассчитаю общую сумму затрат на оплату труда по формуле (5.5):

$$Z_{тр} = 2000 \times 20 + 1000 \times 100 = 140000 (\text{тенге}).$$

В таблице 4.5 приведена амортизация основных фондов.

Таблица 4.5 - Амортизация основных фондов

Наименование оборудования	Стоимость оборудования	Годовая норма амортизации, %	Время работы оборудования для разработки ПП	Сумма, тг
Компьютер	120000	3,7	27	4444,44
Рабочая станция	36200	3,7	27	1347,74
Кондиционер	22000	10	10	2200
ИТОГО амортизация основных фондов				7992,18

При использовании равномерного метода расчета амортизации я получу:

Годовая норма амортизации = (Ежегодные отчисления/ Стоимость оборудования)*100%

Ежегодные отчисления = Стоимость оборудования/ Время работы оборудования для разработки ПП

Ежегодные отчисления (комп.)=120000 / 27 = 4444,44 тг.

Ежегодные отчисления (раб. стан.)=36200 / 27 = 1347,74 тг.

Ежегодные отчисления (кондиц.)=22000 / 10 = 2200 тг.

Годовая норма амортизации (комп.)=(4444,44 / 120000) * 100% = 3,7%

Годовая норма амортизации (раб. стан.)=(1347,74 / 36200) * 100% = 4%

Годовая норма амортизации (кондиц.)=(2200 / 22000) * 100% = 10%

В таблице 4.6 Приведена смета затрат на разработку ПП.

Таблица 4.6 – Смета затрат на разработку ПП

Статьи затрат	Сумма, тг
Заработная плата персонала	140000
Затраты на материалы и запасные части	420400
Амортизация основных фондов	7992,18
Затраты на электроэнергию	3157
Затраты на рекламную компанию	3000

4.4 Определение возможной (договорной) цены ПП

Величина возможной (договорной) цены ПП должна устанавливаться с учетом эффективности, качества и сроков ее выполнения на уровне отвечающем экономическим интересам заказчика (потребителя) и исполнителя.

Договорная цена (C_d) для прикладных ПП рассчитывается по формуле:

$$C_d = Z_{\text{нпр}}(1 + P/100), \quad (1.9)$$

$$C_d = 574549,18 (1 + 30/100) = 576272,82754 \text{ тенге},$$

где $Z_{\text{нпр}}$ - затраты на разработку ПП (из таблицы 4.6), тг; P - средний уровень рентабельности ПП, % (принимается в размере 20-30% по согласованию с консультантом по экономической части).

Далее определяется цена реализации с учетом налога на добавленную стоимость (НДС), ставка НДС устанавливается законодательно Налоговым Кодексом РК. На 2013 год ставка НДС установлена в размере 12%.

Цена реализации с учетом НДС рассчитывается по формуле:

$$C_p = C_d + C_d * \text{НДС}, \quad (1.10)$$

$$C_p = 574549,18 + 576272,82754 * 0,12 = 643701,9193 \text{ тенге}.$$

4.5 Оценка социально-экономических результатов

Экономическая целесообразность данного проекта будет складываться из количественной и качественной составляющих.

Экономический эффект для разработчиков состоит в улучшении финансового положения как персонально разработчиков, так и предприятия, которое займется реализацией проекта. При успешной реализации данного проекта: во-первых – непосредственно разработчики получают зарплату, которая составит 140000 тенге.

Качественный эффект для разработчиков состоит в том, что это первый опыт выпуска проекта на рынок, на котором разработчик решает вопросы дальнейшего проектирования, такие как целесообразность, эффективность проекта, проектирование основы дальнейшей работы. Также в процессе реализации проекта будут решены вопросы маркетинга, рекламы, освоения рынка программного обеспечения.

Результат разработки ПП определяется количественной характеристикой, достигаемой на основе ПП.

Научный эффект характеризует получение новых научных знаний и отражает прирост информации, предназначенной для внутри научного потребления. Научно-технический эффект характеризует возможность использования результатов разработки ПП в других разработках или опытно-конструкторских работах и обеспечивает получение информации, необходимой для создания новых программ. Экономический эффект характеризуется выраженной в стоимостных показателях экономией живого и общественного труда в общественном производстве, полученной при использовании результатов прикладных ПП. Социальный эффект проявляется в улучшении условий труда, повышении экологических характеристик и т.д. В связи с тем, что мой проект не влечет за собой огромные капиталовложения, следовательно, у него очень короткий срок окупаемости, что является очень выгодным.

Вывод

В данной работе рассматривается экономическая оценка проекта по разработке ПП системы защиты данных на платформе Android и Windows. Приведен расчет всех необходимых затрат, связанных с закупкой оборудования и программного обеспечения; различных затрат на энергопотребление и прочего; расчет эксплуатационных расходов: фонд оплаты труда, социальный налог, амортизационные отчисления.

Экономический эффект складывался из: снижения затрат на эксплуатацию оборудования, снижения затрат на ремонт оборудования, повышения экономической эффективности использования основных средств (компьютеров).

Данный проект был рассчитан как небольшой, не требующий для выполнения его большого офисного помещения, большого количества сотрудников. Большой прибыли от проекта ожидать не стоит, но и не стоит считать, что расходы будут велики. Основные расходы заключаются в покупке оборудования и выплате заработной платы.

5 Безопасность жизнедеятельности

5.1 Анализ условий труда при разработке мобильного приложения

5.1.1 Место расположения и рабочий день помещения, изображенного на рисунке 5.1 расположено в жилом многоквартирном доме по адресу г. Алматы ул. Жандосова, 47. Рабочий день: с 10 до 19:00 в будние дни с перерывом в 1 час.

5.1.2 Количество рабочих мест и категория работ по тяжести

В помещении рабочих мест не предполагается. В соответствии с Приложением 1 СанПиН 2.2.4.548-96 «Гигиенические требования к микроклимату производственных помещений» работа относится к категории Ia, то есть работа с интенсивностью энергозатрат не более 120 ккал/ч.

Разработка приложения осуществляется с использованием компьютерной техники и электронного оборудования.

Продолжительность работ превышает 4 ч. Установлены перерывы по 20 мин каждый через 2 ч после начала работ, через 1,5 ч и 2, 5 ч после обеденного перерыва или же по 5-15 мин через каждый час работы. Общее время перерывов не превышает 60 мин.

При разработке прикладного приложения большую роль играет правильная организация условий труда в рабочем помещении. Так как данная работа связана с длительным нахождением за компьютером, необходимо учесть нормы естественного и искусственного освещения, для обеспечения благоприятных условий работы.

5.1.3 План помещения

Помещение имеет следующие параметры:

- находится на первом этаже одноэтажного здания;
- размеры помещения (комнаты): длина 4 м, ширина 3 м, высота 3м;
- вид светопропускающего материала – стекло листовое, двойное; - вид переплета – стальные, двойные, открывающиеся;
- солнцезащитные устройства – убирающиеся регулируемые жалюзи;
- окно размером 1,5*1,2;
- внутренняя отделка стен – светлая;
- помещение по зрительным условиям работы относится к категории легких работ (легкая физическая, категория Ia, работа производится сидя и не требует физического напряжения);
- искусственное освещение – 2 светильника с двумя люминесцентными лампами.

На рисунке 5.1 изображен план помещения, где 1 – окно, 2 – рабочее место, 3 – дверь, 4 – кондиционер.

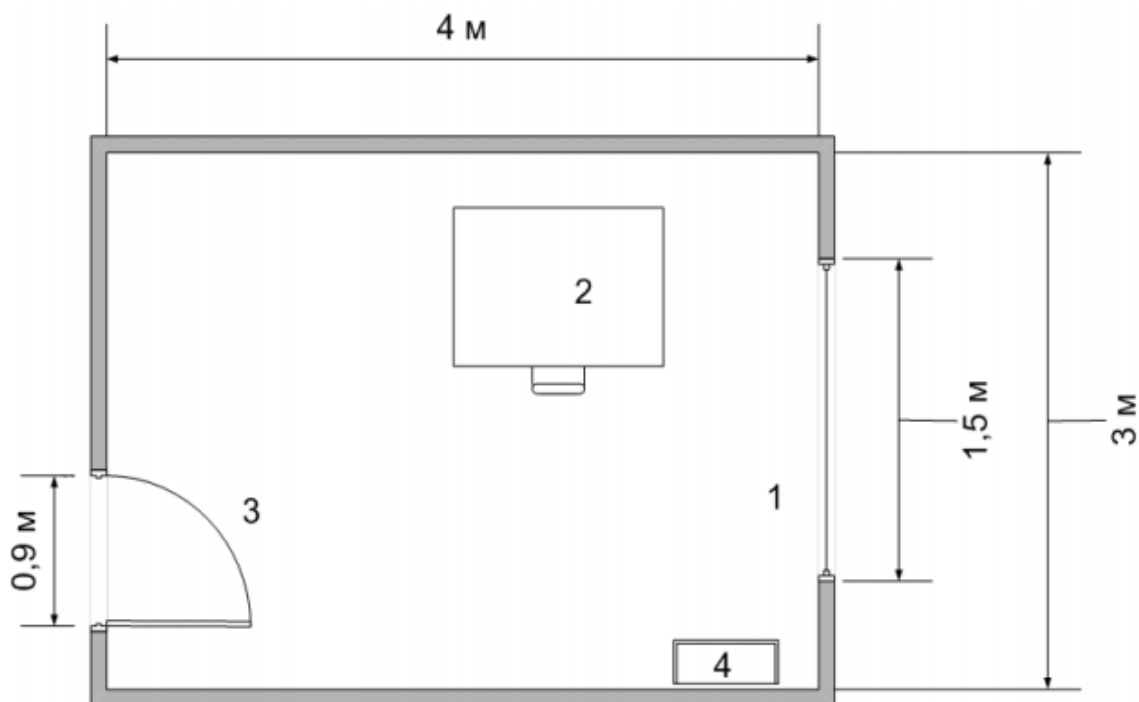


Рисунок 5.1 – План помещения

Характеристики используемого в работе оборудования:

- ноутбук Lenovo Z50, Intel Core i7 3230M, 8 Gb DDR3, 1000 Gb HDD, Radeon HD 7670M;
- электропитание: переменное напряжение 220 – 250 В, частотой 50 Гц и мощностью 90 Вт и 65 Вт соответственно;
- электропитание: переменное напряжение 220 – 250 В, частотой 50 Гц, мощность светильни

Электротехническое оборудование является потенциальным источником возникновения пожарной опасности. Оборудование маломощное – вредность в качестве повышенного шума отсутствует.

Рассматриваемое рабочее помещение расположено в здании, находящемся вдали от железнодорожных путей или нагруженных автомагистралей, аэропорта и так далее, поэтому внешних источников шума, влияющих на процесс работы – нет. Уровень шума соответствует норме согласно ГОСТ 12.1.003-83 ССБТ. Шум. Общие требования безопасности”.

Здание относится к I степени огнестойкости (Здания с несущими и ограждающими конструкциями из естественных или искусственных материалов, бетона или железобетона с применением листовых негорючих материалов). Рабочее помещение по вопросам пожарной безопасности относится к категории “Д”(пониженная пожароопасность в соответствии с ТК РК «Общие требования к пожарной безопасности»). В соответствии с типовыми правилами пожарной безопасности административные здания и отдельные помещения, и технологические установки обеспечиваются первичными средствами пожаротушения согласно нормативам.

5.1.4 Микроклимат

При создании обязательных условий для корректной работы необходимо поддерживать установленные стандартом климатические условия, такие как температура, влажность и скорость движения воздуха. При этом необходимо учесть, что внешние погодные условия не должны влиять на работу. Перечислим ряд требований для работы климатического оборудования:

- поддержание стабильной температуры, чистоты и влажности воздуха;
- обязательное непрерывное поддержание температурного режима круглосуточно;
- наличие в оборудовании системы оповещения о выходе из строя или превышении предельно допустимой температуры.

Стандарт ASHRAE – American Society of Heating, Refrigeration, and AirConditioning Engineers – Американское общество инженеров по нагреванию, охлаждению и кондиционированию воздуха.

Помещение относится к классу A2(серверы, системы хранения данных, персональные компьютеры, рабочие станции со средним уровнем контроля параметров микроклимата). В связи с этим параметры микроклимата согласно ASHRAE от 2011 года указаны в таблице 5.1.

Таблица 5.1 - Параметры микроклимата согласно ASHRAE

Диапазон	Температура по сухому термометру, °C	Диапазон влажности, %	Диапазон точки росы, °C	Максимальная точка росы °C
Фактический	18-27	Не выше 60	5,5-15	15
Допустимый	10-35	20-80	-	21

В помещении установлен постоянно работающий кондиционер марки Samsung модели AR12HQFNAWKNER (рисунок 5.2).

Технические характеристики кондиционера:

- производительность охлаждения – 3 кВт;
- класс энергоэффективности – A/A;
- воздухообмен - 310 м³/час;
- вес внутреннего блока – 11 кг.



Рисунок 5.2 – Кондиционер Samsung AR12HQFNAWKNER

5.1.5 Система освещения

В зависимости от природы источника световой энергии различают естественное, искусственное и совмещенное освещения.

Естественное освещение подразделяют на боковое (одно или двустороннее), когда свет проникает в помещение через световые проемы в наружных стенах; верхнее, осуществляемое через фонари и световые проемы в кровле; верхнее и боковое, сочетающее верхнее и боковое освещения.

Совмещенное освещение применяют в помещениях с недостаточным естественным светом, который дополняется электрическими источниками света, работающими не только в темное, но и в светлое время суток.

Искусственное (электрическое) освещение по характеру выполняемых задач делят на рабочее, аварийное, эвакуационное, охранное и дежурное. Рабочее освещение устраивают во всех помещениях, а также на открытых территориях, предназначенных для работы, прохода людей и движения транспорта. Аварийное освещение предусматривают на случай, когда прекращение или нарушение нормального обслуживания оборудования вследствие выхода из строя рабочего освещения может вызвать пожар, взрыв или отравление людей, длительное нарушение технологического процесса, отказ в работе связи, тепло или электроснабжения, канализации, опасность травмирования, нарушение нормального обслуживания больных. Минимальная освещенность рабочих поверхностей, требующих обслуживания при аварийном режиме, должна быть равна 5% нормируемой освещенности при системе общего освещения. В то же время она не должна быть ниже 2 лк внутри зданий и 1 лк на открытых территориях. Максимальная освещенность при аварийном освещении более 30 лк для газоразрядных и более 10 лк для ламп накаливания требует соответствующего обоснования.

Эвакуационное освещение (аварийное для эвакуации людей) выполняют в местах, опасных для передвижения людей, в основных проходах и на лестничных клетках зданий, в которых работает более 50 чел., или жилых домов в пять этажей и выше, а также в помещениях, выход людей из которых при аварии освещения связан с опасностью травмирования. Наименьшая освещенность на полу, земле или ступенях

должна быть в помещениях 0,5 лк и на открытых территориях 0,2 лк. Для аварийного и эвакуационного освещений разрешается использовать только лампы накаливания, а также люминесцентные лампы в помещениях с температурой воздуха не ниже + 5°C при условии питания ламп напряжением не менее 90% номинального. Светильники аварийного освещения должны отличаться от осветительных приборов рабочего освещения.

Охранное освещение устраивают вдоль границы площадок предприятий, охраняемых в ночное время. При этом освещенность должна быть 0,5 лк на уровне земли в горизонтальной плоскости или в вертикальной плоскости на уровне 50 см от земли. При необходимости часть светильников любого вида освещения можно использовать для дежурного освещения.

В моем помещении применяется как искусственная, так и естественная системы освещения в зависимости от времени суток. Естественное освещение обеспечивается окном размером 1,50м х 1,50м. Для организации искусственного освещения применяется общее освещение, состоящее из двух светильников, расположенных равноудалено.



Рисунок 5.3 – Светильник общего освещения

Аварийного, дежурного освещений не предусмотрено.

5.1.6 Пожарная безопасность

Система пожарной безопасности - это комплекс методов и средств автоматического пожаротушения и оповещения о пожаре, предназначенных для обнаружения пожара и его самоликвидации в начальной стадии.

Системы пожаротушения разрабатываются на основе требований к пожарной безопасности для данной категории помещения. Самым вероятным и распространенным сценарием возникновения пожара является возгорание кабелепроводов из-за перегрева кабелей или компонентов, что в свою очередь приводит к возгоранию кабельной

изоляции. В зависимости от условий период между перегревом и самим возгоранием может варьироваться от нескольких часов до нескольких недель.

Особенностью ликвидации пожара является то, что в серверных комнатах присутствуют перегородки, шкафы особой конструкции и стойки, которые могут стать препятствием для подачи огнетушащего вещества, что в свою очередь, приведет к использованию объемного способа пожаротушения. Огнетушащие вещества не должны оказывать вредного воздействия ни на работу оборудования, ни на обслуживающий персонал.

5.1.7 Электробезопасность

Электробезопасность – совокупность организационных, технических, правовых санитарно-гигиенических, социально-экономических мер, применяемых для защиты людей от воздействия электрического тока.

Система заземления подразумевает решение таких задач, как выравнивание электрических потенциалов и создание низкоомного токопроводящего пути. Система заземления осуществляется согласно следующим принципам:

- создание в соответствии с определенным проектом;
- обязательное оснащение собственной системой заземления для каждой стойки;
- обязательное соединение всех стоек с магистралью заземления;
- обязательное соединение всех металлических деталей к системам заземления;
- обеспечение электрической целостности всех компонентов.

5.2 Расчеты

5.2.1 Расчет искусственного освещения

Существует три метода расчета искусственного освещения:

- точечный метод;
- метод удельной мощности;
- метод коэффициента использования светового потока.

В данной работе будет использоваться метод коэффициента использования светового потока, так как в обоих помещениях светильники расположены равномерно.

Для расчета необходимо установить тип светильника в зависимости от класса помещения и условий среды. В обоих помещениях используются светильники с люминесцентными лампами.

Отношение светового потока, попадающего на расчетную поверхность ко всему потоку, излучаемому светильниками, установленными в помещении, называется коэффициентом использования светового потока в осветительной установке (5.1).

$$N = \frac{F_n + F_{отр}}{n * F_{л}} = \frac{F_y}{n * F_{л}} \quad (5.1)$$

где F_n – световой поток, падающий на освещаемую поверхность, лм;
 $F_{отр}$ – отражённый световой поток от освещаемой поверхности, лм;
 $F_{л}$ – световой поток каждой из ламп, лм;
 N – общее число светильников в помещении;
 n – количество ламп в светильнике.
 Расчетный поток светильников определяется по формуле (5.2):

$$F = \frac{E_h * S * k * z}{\eta}, \quad (5.2)$$

где E_h - выбранная нормируемая освещенность, лм;
 S - площадь помещения, м²;
 k - коэффициент запаса;
 z - отношение средней освещенности к минимальной;
 η - коэффициент использования светового потока ламп, зависящий от типа светильника, коэффициентов отражения потолка и стен, и индекса помещения i .

Индекс помещения рассчитывается по формуле (5.3):

$$i = \frac{S}{h * (A + B)}, \quad (5.3)$$

где A, B - длина и ширина помещения;
 S - площадь помещения;
 h - высота подвеса светильника.

В соответствии с методическими указаниями $\rho_n = 70\%$ (Побеленный потолок: побеленные стены с окнами, закрытыми белыми шторами) и $\rho_c = 50\%$ (Побеленные стены при незавешенных окнах; побеленный потолок в сырых помещениях, чистый бетонный и светлый деревянный потолок). По формуле (5.3) рассчитаем индекс помещения и получим $i = 0,6$.

Теперь можно вычислить значение коэффициента η из таблицы 5.3.

Таким образом, $\eta = 0,22$ [12].

Таблица 5.3 - Значения коэффициента использования светового потока в зависимости от индекса помещения и коэффициентов отражения потолка и стен

Светильник	ПВЛ, $\eta*100$		
$\rho_{\text{п}}, \%$	30	50	70
$\rho_{\text{с}}, \%$	10	30	50
I	$\eta*100$		
0,5	14	16	19
0,6	18	20	22
0,7	21	23	25
0,8	23	25	27
0,9	25	27	29
1,0	26	28	30
1,1	27	29	31
1,25	29	30	32
1,5	30	31	34
1,75	31	33	35
2,0	33	34	36
2,25	34	35	37
3,0	36	37	40
3,5	37	38	40
4,0	38	39	41
5,0	39	40	42

Так как в помещениях используются люминесцентные лампы коэффициент $z=1,1$, а коэффициент $k=1,3$.

Световой поток найдем, рассчитав по формуле (5.2):

$$F = \frac{500*1.1*1.3*12}{0.22} = 39000 \text{ лк} \quad (5.2)$$

Используем лампы ЛДЦ-80 со световым потоком 3560 лм. Тогда общее количество светильников в помещении должно быть рассчитано по формуле (5.3):

$$N = \frac{F}{n*F_{\text{л}}} \quad (5.3)$$

$$N = \frac{39000}{2*3560} = 5,47$$

Таким образом, в помещении количество светильников должно быть 6 шт.

5.2.2 Расчет естественной освещенности

Результатом расчета естественного освещения площадь световых проемов помещений. Для помещений с боковым освещением используется формула (5.4). При верхнем освещении используется формула (5.5).

$$100 * \frac{S_0}{S_n} = \frac{E_n * K_3 * \eta_0}{\tau_0 * r_1} * K_{3Д} \quad (5.4)$$

$$100 * \frac{S_0}{S_n} = \frac{E_n * K_3 * \eta_\phi}{\tau_0 * r_1 * K_\phi} \quad (5.5)$$

где S_0 - площадь световых проемов, m^2 ;
 S_n - площадь пола помещения, m^2 ;
 E_n - нормируемое значение КЕО, m^2 ;
 K_3 - коэффициент запаса;
 η_0 - световая характеристика окон;
 τ_0 - общий коэффициент светопропускания, где τ_0 рассчитывается по формуле (5.6);
 r_1 - коэффициент, учитывающий повышение КЕО при боковом освещении;
 $K_{3Д}$ - коэффициент, учитывающий затемнение окон;
 S_ϕ - площадь световых проемов;
 r_2 - коэффициент, учитывающий повышение КЕО при верхнем освещении;
 K_ϕ - коэффициент, учитывающий тип фонаря.

$$\tau_0 = \tau_1 * \tau_2 * \tau_3 * \tau_4 * \tau_5 \quad (5.6)$$

где τ_1 - коэффициент светопропускания материала;
 τ_2 - коэффициент, учитывающий потери света в переплетах светопроема;
 τ_3 - коэффициент, учитывающий потери света в несущих конструкциях, при боковом освещении равен 1;
 τ_4 - коэффициент, учитывающий потери света в солнцезащитных устройствах;
 τ_5 - коэффициент, учитывающий потери света в защитной сетке, устанавливаемой под фонарями.

Так как в помещении используется боковое освещение, расчет будет проводиться по формуле (5.4).

Нормируемое значение КЕО для соответствующего района

определяется по формуле (5.7).

$$e_N = E_H * m_N \quad (5.7)$$

где N – номер группы административных районов;

E_H - значение КЕО;

m_N - коэффициент светового климата.

Так как противостоящие здания находятся на расстоянии более 3 метров, $K_{зд} = 1$.

Значение коэффициента запаса K_3 для помещений общественных и жилых зданий при вертикально расположенном светопропускающем материале равен 1,2.

Посчитаем значение τ_0 по формуле (5.7) учитывая значения коэффициентов из методических указаний, и получим $\tau_0 = 0,36$.

Посчитаем значение E_N по формуле (5.8) для г. Алматы получим $E_N = 1.5 * 0.65 = 0.975$.

Значение световой характеристики для помещения $\eta_0 = 8$.

Имея необходимые данные, рассчитаем площадь световых проемов по формуле (5.4).

$$100 * \frac{S_0}{12} = \frac{0.975 * 1.2 * 8}{0.36 * 1.1} * 1$$

Получим $S_0 = 2.83 \text{ м}^2$.

5.2.3 Расчет вентиляции

Для расчета необходимого количества воздуха используется формула (5.8).

$$L = n * V (\text{м}^3 / \text{час}) \quad (5.8)$$

где n - нормируемая кратность воздухообмена, час^{-1} ;

V - объём помещения, м^3 .

Нормируемая кратность воздухообмена есть отношения объема воздуха поступающего в помещение в течение часа к объему помещения [13].

Таблица 5.12 - Нормируемая кратность воздухообмена

Помещение	Кратность воздухообмена
Офисное помещение	5-7

Кратность воздухообмена для нашего помещения выберем = 7.

Теперь рассчитаем необходимый объем количества используемого воздуха по формуле 5.9.

$$L = 7 \cdot 4 \cdot 3 \cdot 3 = 252 \text{ м}^3/\text{час}.$$

Вывод

В конце подведу итоги проделанных расчетов и исследований, а именно:

- соблюдение стандартов поддержания микроклимата в помещении;
- соблюдение стандартов по вентиляционным параметрам;
- соблюдение норм освещения;
- соблюдение норм безопасности с электроприборами и пожарной безопасности.

По результатам расчетов была спроектирована система освещения для рабочей поверхности: выбран тип светильников, а также их количество и расположение.

Были произведены расчеты в помещении для системы вентиляции. Исходя из расчетов, в помещении необходимо было установить один кондиционер.

Заключение

Во время выполнения дипломного проекта была разработана система защиты данных на платформе Android и Windows.

Были выполнены следующие задачи:

- проведен анализ платформы для разработки программных продуктов;
- рассмотрены модели криптографической защиты данных на платформе Android и Windows;
- рассмотрены алгоритмы шифрования;
- выдвинут свой алгоритм шифрования;
- реализована система защиты данных;
- рассчитаны экономические расходы использования программы;
- описана безопасность жизнедеятельности;
- сделаны соответствующие выводы и заключения.

В первой главе были рассмотрены методы шифрования, выдвинут собственный алгоритм шифрования, внедренный в приложение, а также описан принцип работы каждого из алгоритмов.

Во второй главе рассмотрены операционная система Android, ее история, преимущества и недостатки, архитектура, а также инструменты для разработки и отладки приложений, их пошаговая установка, и компоненты Android-приложений.

В третьей главе описан процесс создания приложения на ОС Android в среде разработки Android Studio и программы на ОС Windows в среде разработки Microsoft Visual Studio, их структура, а также структура каталогов и файлов ресурсов. Описан процесс подключения абонентов в мессенджере на Android, переписка и защита данных в приложении. В программе на Windows показан процесс шифрования и дешифрования текста.

Описан весь функционал приложений:

- два метода шифрования: AES, RSA;
- защита приложений;
- защита данных при их отправке другому абоненту;
- шифрование и дешифрование текста.

Четвертая глава посвящалась технико-экономическому обеспечению данного дипломного проекта. Здесь рассчитывались затраты на проектирование и разработку программного продукта.

Анализ вредных факторов трудового процесса приведён в пятой главе. Так же здесь произведен расчет обеспечения оптимальной освещенности помещения и для системы вентиляции. Исходя из полученных данных установлен кондиционер.

Список использованных сокращений

НСД – несанкционированный доступ.

ОС - операционная система.

ПО - программное обеспечение.

ПП - программный продукт.

СУБД – система управления базами данных – это совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием базы данных.

КЕО – коэффициент естественной освещенности.

Список литературы

1. С. П. Панасенко, В. П. Батура. Основы криптографии для экономистов: Учебное пособие. – М.: 2005.
2. А. Н. Курев. Защита персональных данных с помощью алгоритмов шифрования. URL: <http://referatwork.ru/refs/source/ref-10578.html/>.
3. Б. Анин. Защита компьютерной информации. - СПб.: БХВ-Петербург, 2008.
4. Алферов А. Основы криптографии. Учебное пособие. - СПб.: БХВ-Петербург, 2002.
5. О. А. Логачёв, А. А. Сальников, С. В. Смышляев, В. В. Ященко. Булевы функции в теории кодирования и криптологии. – М.: 2012.
6. Яковлев А. А., Безбогов В. В., Родин В. Н. Криптографическая защита информации: учебное пособие. - Тамбов: Изд-во Тамб. гос. техн. ун-та, 2006.
7. П. Н. Аккотов. Android. URL: <https://ru.wikipedia.org/wiki/Android/>.
8. Голощапов А.Л. Google Android программирование для мобильных устройств. - BHV Санкт-Петербург, 2005.
9. Роджерс Рик, Ломбардо Джон, Медниекс Зигурд, Мейк Блейк. Android. Разработка приложений. – ЭКОМ Паблишерз, 2010.
10. Колисниченко Д. Программирование для Android. Самоучитель. - СПб.: Санкт-Петербург, 2011.
11. Аманжолова К. Б., Алибаева С. А. Экономика предприятия телекоммуникации: Учебное пособие. - Алматы: АИЭС, 2003.
12. Корольченко А.В. Естественное и искусственное освещение. - М.: Издательство Москва, 2004.
13. Дюсебаев М.К. Безопасность жизнедеятельности: методические указания к выполнению раздела дипломных проектов. – Алматы: АИЭС, 2003.

Приложение А

Класс MainActivity

```
package kz.diplom.myapplock;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.Views.View;
import android.widget.Button;
public class MainActivity extends AppCompatActivity {
    Button bShifr, bProtect, bSettings, bAbout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bShifr = (Button) findViewById(R.id.bShifr);
        bProtect = (Button) findViewById(R.id.bProtect);
        bSettings = (Button) findViewById(R.id.bSettings);
        bAbout = (Button) findViewById(R.id.bAbout);
        bShifr.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, Shifr.class));
            }
        });
        bProtect.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this,
                Protect.class));
            }
        });
        bSettings.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, Settings.class));
            }
        });
        bAbout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(MainActivity.this, About.class));
            }
        });
    }
}
```

Класс Shifr

```
package kz.diplom.myapplock;
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.SparseBooleanArray;
import android.Views.Menu;
import android.Views.MenuItem;
import android.Views.Views;
import android.widget.Button;
import android.widget.Toast;
import java.util.HashSet;
import java.util.Set;
public class Shifr extend AppCompatActivity implements
    Views.OnClickListener{
    Button bShifrBase64,bShifrAES,bShifrRSA,bShifrDES;
    @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_shifr);
            bShifrBase64 = (Button) findViewById(R.id.bShifrBase64);
            bShifrBase64.setOnClickListener(this);
            bShifrAES = (Button) findViewById(R.id.bShifrAES);
            bShifrAES.setOnClickListener(this);
            bShifrRSA = (Button) findViewById(R.id.bShifrRSA);
            bShifrRSA.setOnClickListener(this);
            bShifrDES = (Button) findViewById(R.id.bShifrDES);
            bShifrDES.setOnClickListener(this);
        }
        @Override
        public void onClick(Views Views) {
            switch (Views.getId()){
                case R.id.bShifrBase64:
                    startActivity(new Intent(Shifr.this,Base.class));
                    break;
                case R.id.bShifrAES:
                    startActivity(new Intent(Shifr.this,AES.class));
                    break;
                case R.id.bShifrRSA:
                    startActivity(new Intent(Shifr.this,RSAA.class));
                    break;
                case R.id.bShifrDES:
```

Класс Base

```
package kz.diplom.myapplock;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Base64;
import android.Views.Menu;
import android.Views.MenuItem;
import android.Views.Views;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import java.io.UnsupportedEncodingException;
public class Base extends AppCompatActivity implements
Views.OnClickListener{
    Button bBase64E,bBase64D,bBaseCopy;
    EditText tBase64;
    TextView rBase64;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_base);
        bBase64D = (Button) findViewById(R.id.bBase64D);
        bBase64D.setOnClickListener(this);
        bBase64E = (Button) findViewById(R.id.bBase64E);
        bBase64E.setOnClickListener(this);
        bBaseCopy = (Button) findViewById(R.id.bBaseCopy);
        bBaseCopy.setOnClickListener(this);
    }
    @Override
    public void onClick(Views Views) {
        tBase64 = (EditText) findViewById(R.id.tBase64);
        rBase64 = (TextView) findViewById(R.id.rBase64);
        String s = tBase64.getText().toString();
        switch (Views.getId()){
            105
            case R.id.bBaseCopy:
                tBase64.setText(rBase64.getText().toString());
                break;
            case R.id.bBase64D:
                byte[] result = Base64.decode(s,0);
                rBase64.setText(new String(result));
```



```

        break;
        case R.id.bBase64E:
            String result2 = null;
            try {
                result2 = Base64.encodeToString(s.getBytes("UTF-8"), 0);
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            }
            rBase64.setText(result2);
            break;
        }
    }
    @Override
    public void onBackPressed() {
        startsActivity(new Intents(Base.this, Shifr.class));
    }
}

```

Класс AES

```

package kz.diplom.myapplock;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Base64;
import android.Views.Menu;
import android.Views.MenuItem;
import android.Views.Views;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
public class AES extend AppCompatActivity implements
    Views.OnClickListener{
    Button bAESE,bAESD,bAESCpy;
    EditText tAES;
    TextView rAES;
    @Override
    106
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_aes);
    }
}

```

```

        bAESE = (Button) findViewById(R.id.bAESE);
        bAESE.setOnClickListener(this);
        bAESD = (Button) findViewById(R.id.bAESD);
        bAESD.setOnClickListener(this);
        bAESCopY = (Button) findViewById(R.id.bAESCopY);
        bAESCopY.setOnClickListener(this);
    }

    public static String encrypt(String key, String initVector, String value) {
    try {

        IvParameterSpec iv = new
        IvParameterSpec(initVector.getBytes("UTF-8"));
        SecretKeySpec skeySpec = new
        SecretKeySpec(key.getBytes("UTF-8"), "AES");
        Cipher cipher =
        Cipher.getInstance("AES/CBC/PKCS5PADDING");
        cipher.init(Cipher.ENCRYPT_MODE, skeySpec, iv);
        byte[] encrypted = cipher.doFinal(value.getBytes());
        System.out.println("encrypted string: "
        + Base64.encodeToString(encrypted, 0));
        return Base64.encodeToString(encrypted, 0);
    } catch (Exception ex) {
        ex.printStackTrace();
    }

    return null;
    }

    public static String decrypt(String key, String initVector, String encrypted)
    {
    try {

        IvParameterSpec iv = new
        IvParameterSpec(initVector.getBytes("UTF-8"));
        SecretKeySpec skeySpec = new
        SecretKeySpec(key.getBytes("UTF-8"), "AES");
        Cipher cipher =
        Cipher.getInstance("AES/CBC/PKCS5PADDING");
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, iv);
        byte[] original = cipher.doFinal(Base64.decode(encrypted, 0));
        return new String(original);
    } catch (Exception ex) {
        ex.printStackTrace();
    }

    return null;
    }
    @Override

```

Класс RSAA

```
package kz.diplom.myapplock;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Base64;
import android.Views.Menu;
import android.Views.MenuItem;
import android.Views.Views;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.security.InvalidKeyException;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.Security;
import javax.crypto.Cipher;
import javax.crypto.NoSuchPaddingException;
public class RSAA extend AppCompatActivity implements
Views.OnClickListener{
    Button bRSAE,bRSAD,bRSACopy;
    EditText tRSA;
    TextViews rRSA,tPubKey,tPriKey;
    PublicKey publicKey;
    PrivateKey privateKey;
    Cipher cipher;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_rsa);
    bRSAE = (Button) findViewById(R.id.bRSAE);
    bRSAE.setOnClickListener(this);
    bRSAD = (Button) findViewById(R.id.bRSAD);
    bRSAD.setOnClickListener(this);
    bRSACopy = (Button) findViewById(R.id.bRSACopy);
    bRSACopy.setOnClickListener(this);
try {
    cipher = Cipher.getInstance("RSA");
    KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
```

```

keyGen.initialize( 2048 );
KeyPair kp = keyGen.genKeyPair();
publicKey = kp.getPublic();
privateKey = kp.getPrivate();
tPubKey = (TextViews) findViewById(R.id.tPubKey);
tPriKey = (TextViews) findViewById(R.id.tPriKey);
tPubKey.setText("Публичный ключ: " + publicKey.toString());
tPriKey.setText("Частный ключ: " + privateKey.toString());
} catch (Exception e) {
    e.printStackTrace();
}
}
@Override
public void onClick(Views Views) {
    tRSA = (EditText) findViewById(R.id.tRSA);
    rRSA = (TextViews) findViewById(R.id.rRSA);
    String s = tRSA.getText().toString();
    switch (Views.getId()){
        case R.id.bRSACopy:
            tRSA.setText(rRSA.getText().toString());
            109
            break;
        case R.id.bRSAE:
            try {
                cipher.init( Cipher.ENCRYPT_MODE, publicKey );
                byte[] x = cipher.doFinal( s.getBytes() );
                String str = Base64.encodeToString(x, 0);
                rRSA.setText(str);
            } catch (Exception e) {
                e.printStackTrace();
            }
            break;
        case R.id.bRSAD:
            try {
                cipher.init(Cipher.DECRYPT_MODE, privateKey);
                byte[] str = Base64.decode(s, 0);
                byte[] plainText = cipher.doFinal(str);
                rRSA.setText(new String(plainText));
            } catch (Exception e) {
                e.printStackTrace();
            }
            break;
    }
}

```

```

        @Override
        public void onBackPressed() {
            startsActivity(new Intent(RSAA.this, Shifr.class));
        }
    }
}

```

Приложение Б

Листинг программы на Windows

```

unit Unit1;
interface
uses
    Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
    Dialogs, StdCtrls, ExtCtrls, RsaOpenSSL, ImgList, ComCtrls, ToolWin,
    Menus;
type
    TForm1 = class(TForm)
        MainMenu1: TMainMenu;
        N1: TMenuItem;
        N2: TMenuItem;
        N3: TMenuItem;
        N4: TMenuItem;
        ToolBar1: TToolBar;
        ToolButton1: TToolButton;
        ToolButton2: TToolButton;
        ToolButton3: TToolButton;
        ToolButton4: TToolButton;
        ToolButton5: TToolButton;
        ToolButton10: TToolButton;
        ToolButton11: TToolButton;
        ImageList1: TImageList;
        GroupBox1: TGroupBox;
        GroupBox2: TGroupBox;
        GroupBox3: TGroupBox;
        GroupBox4: TGroupBox;
        Memo1: TMemo;
        Memo2: TMemo;
        Memo3: TMemo;
        Memo4: TMemo;
        SHA2561: TMenuItem;
        SHA5121: TMenuItem;
        N5: TMenuItem;
    end;

```

```

GroupBox5: TGroupBox;
Memo5: TMemo;
ToolButton12: TToolButton;
ToolButton13: TToolButton;
procedureFormCreate(Sender: TObject);
procedure ToolButton2Click(Sender: TObject);
procedure ToolButton4Click(Sender: TObject);
procedure ToolButton10Click(Sender: TObject);
procedure N4Click(Sender: TObject);
procedure SHA2561Click(Sender: TObject);
procedure SHA5121Click(Sender: TObject);
procedure N5Click(Sender: TObject);
procedure N3Click(Sender: TObject);
procedure N2Click(Sender: TObject);
procedure ToolButton12Click(Sender: TObject);
private
{ Private declarations }
fRSAOpenSSL :TRSAOpenSSL;
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
{$R *.dfm}
uses unit4,unit2;
var
aAESPPathToPublicKey, aAESPPathToPrivateKey: string;

procedure TForm1.FormCreate(Sender: TObject);
var
aPathToPublicKey, aPathToPrivateKey: string;
begin
aPathToPublicKey := 'public.pem';
aPathToPrivateKey := 'private.pem';
fRSAAESOpenSSL := TRSAAESOpenSSL.Create(aPathToPublicKey,
aPathToPrivateKey);
end;

procedure TForm1.ToolButton2Click(Sender: TObject);
var
aRSAAESData: TRSAAESData;
n,i: integer;
begin

```

```

randomize;
n:=random(256);
aAESPPathToPublicKey:="";
for i:=1 to n do
aAESPPathToPublicKey:=aAESPPathToPublicKey+char(random(224)+32);
randomize;
  n:=random(256);
aAESPPathToPrivateKey:="";
for i:=1 to n do
aAESPPathToPrivateKey:=aAESPPathToPrivateKey+char(random(224)+32);

aRSAAESData.DecryptedData := Memo1.Text;
fRSAAESOpenSSL.PublicKeyEncrypt(aRSAAESData);
ifaRSAAESData.ErrorResult = 0 then
memo2.Lines.Text := aRSAAESData.CryptedData;
memo4.Lines.Add(aRSAAESData.ErrorMessage);
memo4.Lines.Add('Шифрованиеключ ='
+fRSAAESOpenSSL.SHA1_base64(aAESPPathToPublicKey));
memo4.Lines.Add('Дешифрованиеключ ='
+fRSAAESOpenSSL.SHA1_base64(aAESPPathToPrivateKey));
end;

```