**МИНОБРНАУКИ РОССИИ**

**Федеральное государственное автономное образовательное учреждение**

**высшего профессионального образования**

**«Национальный исследовательский университет «МИЭТ»**

**Факультет** Микроприборов и технической кибернетики (МПиТК)

**Кафедра «**Телекоммуникационные системы»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**НА ТЕМУ:**

**Разработка программно-аппаратной системы двухфакторной аутентификации»**

**Студент /Р.В. Иванов /**

**(подпись)**

**Руководитель /В. П. Батура/**

**(подпись)**

**Консультант /О. П. Симонова/**

**(подпись)**

**2015 г.**

# Техническое задание

* 1. **Назначение и область применения**.Разработка программно-аппаратной системы двухфакторной аутентификации, предназначенная для доступа к настройкам оборудования. Предназначено для безопасной аутентификации.
  2. **Цель работы:**

1. разработать систему двухфакторной аутентификации, на основе российских криптографических стандартах;
2. научиться разрабатывать сетевые приложения и криптографические протоколы для различного назначения;
3. освоить современные методы разработки программного обеспечения.
   1. **Задачи проекта:**
4. реализовать библиотеку, содержащий алгоритм симметричного шифрования ГОСТ 28147-89, вычисления функций хеширования ГОСТ Р 11.34-2012 и протокол обмена ключами шифрования Диффи – Хеллмана для ОС Windows Server 2012 и Linux;
5. разработать криптографический протокол для двухфакторной аутентификации, согласно требованиям технического задания;
6. написать приложения для тестирования библиотек;
7. написать документ, описывающий интерфейс к библиотекам.
   1. **Требования к платформе:**
8. программное обеспечение для сервера: ОС Windows Server 2012R2
9. программное обеспечение для клиента: ОС Linux (встраиваемая система)
   1. **Требования к криптографическому протоколу:**
10. применять только сертифицированные российские стандарты;
11. соответствовать требованию ФСТЭК для информационных систем 3 класса;
12. протокол должен содержать только необходимый для данной задачи функционал.
    1. **Результат работы:**
13. библиотека, включающая в себе реализацию применяемых криптографических примитивов, работающая под выше сказанные платформы;
14. библиотека, реализующая криптографический протокол связи клиента с сервером;
15. библиотека, реализующая протокол взаимодействия смарт-карты и клиента, через USB интерфейс;
16. приложение для тестирования библиотек;
17. документ, описывающий интерфейс к библиотекам.
    1. **Требования к обеспечению надёжного функционирования программ.**

Надежное (устойчивое) функционирование программных обеспечений должно быть обеспечено выполнением совокупности организационно-технических мероприятий. Перечень организационно – технических мероприятий:

1. использование во время разработки программу контроля версий;
2. использованием лицензионного программного обеспечения для разработки:
3. Microsoft visual studio 2012;
4. ОС Linux OPENSUSE 13.1;
5. QtCreater.
6. регулярное выполнения ГОСТ 51188-98 защита информации, испытание программных средств на наличие программных вирусов.
   1. **Требования к программной документации.**

Программная документация содержит:

1. техническое задание;
2. программы и методики испытаний;
3. описание интерфейсных функций библиотек:
4. криптографических алгоритмов;
5. криптографического протокола;
6. протокола взаимодействия со смарт-картой по USB интерфейсу;
7. описание приложения для тестирования библиотек:
8. руководств по применению библиотек.
   1. **Стадии разработки.**

Разработка должна быть проведена в 4 стадии:

1. разработка технического задания;
2. разработка программных обеспечений;
3. тестирование по методикам испытаний:
4. внедрение продуктов.
   1. **Этапы разработки.**

Разработка должна быть проведена в нескольких этапах. На стадии разработки технического задания должно быть поставлены основные требования к системе:

1. функциональные возможности ПО;
2. требования к аппаратным составляющим;
3. требования к защите информации и криптографическому протоколу;
4. сроки разработки.

В процессе разработки должны быть проведены следующие основные работы:

1. разработка архитектуры программного обеспечения;
2. реализация криптографических алгоритмов;
3. разработка приложения тестирования криптографических алгоритмов;
4. разработка криптографического протокола;
5. разработка приложений для тестирования криптографического протокола;
6. написание программной документации;
   1. **Порядок контроля. Статический анализ исходного кода.**

Программный код должна быть исследован с помощью статистического анализа,с целью выявления возможных ошибок, а так же потенциально небезопасного кода на этапе написания программы.

# ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

БД – база данных

ОС – операционная система

ПО – программное обеспечение

ИТ – информационные технологии

СОДЕРЖАНИЕ

[Техническое задание 2](#_Toc419661378)

[ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 5](#_Toc419661379)

[ВВЕДЕНИЕ 7](#_Toc419661380)

[ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ 9](#_Toc419661381)

[Обзор современных методов аутентификации. 9](#_Toc419661382)

[1.1. Основные понятия 9](#_Toc419661383)

[1.2. Однофакторная аутентификация. Основные недостатки и методы атак на систему. 12](#_Toc419661384)

[1.2.1. Полный перебор паролей. 14](#_Toc419661385)

[1.3. Двухфакторная аутентификация. Преимущества и недостатки. 20](#_Toc419661386)

[Криптографические протоколы аутентификации. 22](#_Toc419661387)

[2. 1. Аутентификация клиента и сервера методом «запрос - ответ». 24](#_Toc419661388)

[2.2. Сравнение протоколов аутентификации клиента. 28](#_Toc419661389)

[2.3. Протокол обмена сессионными ключами. 29](#_Toc419661392)

[1. Описание разрабатываемого протокола. 31](#_Toc419661393)

# ВВЕДЕНИЕ

Аутентификация является важным элементом в обеспечении безопасности информационных и инфокоммуникационных систем от несанкционированного доступа Проверка подлинности идентификатора, предъявленного пользователем, является одним из сложных и актуальных задач в обеспечении информационной безопасности инфокоммуникационных систем.

Процесс регистрации пользователя в системе состоит из трёх взаимосвязанных последовательных процедур: идентификация, аутентификация, авторизация(1).

Идентификация – это процедура распознавания субъекта по идентификатору.

Аутентификация – это процедура проверки подлинности субъекта, позволяющая достоверно убедиться в том, что субъект, предъявляющий свой идентификатор, является именно тем, идентификатор которого он использует. Для этого он должен подтвердить факт обладания некоторой информацией, которая может быть доступна только ему одному.

Авторизация – это процедура предоставления пользователю определённых прав доступа к ресурсу, после прохождения процесса аутентификации.

Говоря о факторах аутентификации, следует выделить основные три вида:

1. на основе знаний чего – либо (пароль, парольная фраза или ПИН);
2. на основе обладаний чем – либо (физический ключ, карта с магнитной полосой, USB токен, генерирующий одноразовый пароль);
3. на основе биометрии.

Аутентификация, в процессе которой используется только один тип аутентификационных факторов, называется однофакторной, при использовании нескольких аутентификационных факторов - многофакторной аутентификацией. Самым простым и распространенным видом однофакторной аутентификации является парольная аутентификация. Такой методпоявился одним из первыхв ИТ-системах, реализующих множественный доступ к информационным ресурсам. Каждый субъект имеет пароль, секретная информация, которая доступна только ему. Такой метод имеет ряд преимуществ:

* простота применения;
* несложная реализация;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Статься «Методы и средства аутентификации в задачах обеспечения информационной безопасности в корпоративных информационных системах» О. В. Куликова

Однако, парольная аутентификация является самым уязвимым в ИТ – системе, так как пароль легче всего украсть, подобрать и подсмотреть злоумышленником. Что говорит о недостатках парольной аутентификации. Подтверждением этому является статья опубликованная компанией RSA: «Why password aren’tstrongenough». В этой статье показаны результаты исследований надежности паролей пользователей.

Другим распространенным методом является двухфакторная аутентификация, которая основана на основе обладании чем – либо и на основе знании чего либо. В таких системах, кроме пароля используются аппаратные средства, которые хранят и обрабатывают конфиденциальные данные о пользователе. Примером являются:

* электронные ключи (Touch Memory);
* смарт-карты (магнитные карты или пластиковые карты со встроенным микропроцессором);
* USB – ключи.

Двухфакторная аутентификация является на сегодняшний день наиболее распространенным и оправдывающим своё применение.

# ИССЛЕДОВАТЕЛЬСКИЙ РАЗДЕЛ

## Обзор современных методов аутентификации.

## Основные понятия

*Идентификация* – процедура распознавания субъекта (пользователя, процесса, действующего от имени пользователя, аппаратно-программного компонента) по его уникальному идентификатору, присвоенному субъекту ранее и занесенному в базу данных в момент его регистрации как легального пользователя системы.

*Аутентификация* – процедура проверки подлинности входящего в систему субъекта, предъявившего свой идентификатор.

*Авторизация* – это процедурой предоставления субъекту определённых прав.

В любой системе аутентификации обычно можно выделить следующие элементы:

* субъект, проходящий процедуру аутентификации;
* характеристика субъекта (отличительная черта);
* система, несущая ответственность за аутентификацию и контролирующий её работу (сервер);
* сам механизм аутентификации, то есть принцип работы системы;
* механизм, предоставляющий или лишающий субъекта определенных прав доступа.

Субъект может подтвердить свою подлинность, предъявив, один из следующих аутентификаторов:

1. Аутентификация на основе «субъект знает» – претендент обладает информацией, которой нет y других объектов компьютерной системы (пароль, персональный идентификационный номер, секретный ключ). Эту информацию субъект демонстрирует в протоколах типа «запрос – ответ».
2. Аутентификация на основе «субъект обладает» – претендент имеет некий физический предмет (магнитную карту, интеллектуальную карту, смарт-карту, генератор паролей), который выполняет для него криптографические преобразования информации;
3. Аутентификация на основе «субъект есть» – проверяются некоторые биометрические данные человека (голос, радужная оболочка глаза, отпечатки пальцев и др.).

*Фактор аутентификации* – определенный вид информации, предоставляемый субъектом системе для его аутентификации. Если в процессе аутентификации используется только один способ аутентификации, такая система называется *однофакторной* аутентификацией, если несколько – *многофакторной*. Для идентификации средствами криптографии все эти три метода аутентификации могут быть сведены к одному – к аутентификации на основе владения какой-либо информацией. Действительно, любые биометрические данные или информация, заключенная на физическом носителе, могут быть преобразованы в уникальный ключ (при идентификации при помощи криптографической системы или протокола) или пароль (при аутентификации или идентификации парольными схемами), который будет однозначно определять субъекта.

*Протокол аутентификации* – это криптографический протокол, в ходе которого одна сторона убеждается:

1. в идентичности другой стороны, вовлеченной в протокол;
2. в активности другой стороны во время или непосредственно перед моментом приобретения доказательства.

Аутентификация может быть *односторонней* (когда клиент доказывает свою подлинность серверу) и *двусторонней* или *взаимной* (это обоюдная аутентификация между сторонами обмена информацией).

Примером *односторонней* аутентификации является процедура входа в систему Windows NT.

Примером *двусторонней* является использование протокола Kerberos.

Протоколы аутентификации характеризуются следующими важными характеристиками:

1. *Вычислительная эффективность* количество операций, необходимых для выполнения протокола;
2. *Коммуникационная эффективность* – данное свойство отражает количество сообщений и их длину, необходимую для осуществления аутентификации;
3. *Наличие третьей стороны* – примером третьей стороны может служить доверенный сервер распределения симметричных ключей или сервер, реализующий дерево сертификатов открытых ключей, основа гарантий безопасности – примером могут служить протоколы, обладающие свойством доказательства с нулевым знанием;
4. *Хранения секрета* – имеется в виду способ хранения критичной ключевой информации.

Также, можно классифицировать протоколы аутентификации по уровню обеспечиваемой безопасности или возможности противостоять определенному классу атак.

В соответствии с данным подходом протоколы аутентификации делят на типы:

* простая аутентификация (на основе использования паролей);
* строгая аутентификация (на основе использования криптографических методов и средств);
* протоколы, обладающие свойством доказательства с нулевым знанием.

## Однофакторная аутентификация. Основные недостатки и методы атак на систему.

Один из способов аутентификации в компьютерной системе состоит во вводе пользовательского идентификатора («логина» от англ. login – регистрационное имя пользователя) и пароля конфиденциальной информации, доступной только пользователю. Достоверная (эталонная) пара логина и значение пароля хранится на сервере базы данных.

Пароли бывают фиксированные и однократные.Фиксированные или многократные пароли не меняются при каждом обращении, а однократные используются только один раз. Простая аутентификация имеет следующий алгоритм:

1. Субъект запрашивает доступ в систему и вводит личный идентификатор (логин) и пароль. Поскольку пароль должен храниться в тайне, то он должен шифроваться перед посылкой по незащищенному каналу связи.
2. Идентификатор и пароль поступают на сервер аутентификации, где сравниваются с эталонным идентификатором и паролем. В случае их совпадения, пароль считается достоверным, а пользователь – законным.
3. При совпадении с эталоном, аутентификация признается успешной, при несовпадении – субъект возвращается на первый шаг. При этом пароль и идентификатор могут передаваться по сети следующими способами:

* в открытом виде (например, при доступе к IIS (Internet Information Server) по протоколу НТТР в качестве одного из типов поддерживаемой IIS аутентификации);
* идентификатор пользователя, его пароль вначале шифруются на согласованном ключе, и в зашифрованном виде пересылается системе;
* идентификатор пользователя, его парольвначале подаются на вход односторонней функции (хэш-функции), а выход функции пересылается системе;

Схемы организации простой аутентификации отличаются также способами хранения паролей:

1. Xранений паролей в открытом виде в системных файлах, с установленными на них атрибутами защиты от чтения и записи (например, при помощи описания соответствующих привилегий в списках контроля доступа к операционной системе). Криптографические механизмы (шифрование, однонаправленные функции) не используются.

Недостаток: если злоумышленник получит в системе привилегии администратора, то ему станут доступны системные файлы и ресурсы. Для усиления защиты этого способа хранения паролей можно записать парольные файлы на внешние носители (touch memory, смарт-карты и гибкие магнитные диски). Однако, такой подход так же не гарантирует полную безопасность.

1. Более безопасным методом является хранение паролей в *зашифрованном* виде, или, что удобнее, в *хешированном* виде. При проверке введенного пароля система вычисляет его хeш-образ и сравнивается результатом хеш-образа пароля, который храниться в базе данных, соответствующий логину пользователя. Такой подход хранения более надежный, чем предыдущий.

Всё-таки, главной угрозой *парольной аутентификации* – это взлом пароля, т.е. раскрытие злоумышленником пароля пользователя, дающего право входить в систему.

К основным атакам паролей пользователей относятся:

1. Полный перебор паролей.
2. Атака с помощью словаря.
3. Атака с помощью радужных таблиц.
4. Метод социальной инженерии (основан на предположении, что пользователь использовал в качестве пароля личные сведения – имя или фамилия, дата рождения и т.п.).
5. Установка вредоносных программ для перехвата пароля.
6. Подмена доверенного объекта сети (IP-spoofing).
7. Перехват пакетов (sniffing).

Поэтому, для уменьшения деструктивного влияния человеческого фактора необходимо реализовать ряд требований к выбору и использованию паролей:

1. Задание минимальной длины пароля для затруднения метода полного перебора.
2. Использование в пароле различных групп символов для усложнения подбора пароля.
3. Проверка и отбраковка пароля по словарю для затруднения проведения злоумышленником словарной атаки.
4. Установление максимального срока действия пароля для затруднения метода полного перебора паролей, в том числе и в режиме «off-line» при взломе предварительно похищенных учетных записей пользователей
5. Ограничение числа попыток ввода пароля для предотвращения интерактивного подбора пароля злоумышленником.
6. Использование задержки при вводе неправильного пароля для предотвращения интерактивного подбора пароля злоумышленником (в режиме on-linе).
7. Поддержка режима принудительной смены пароля пользователя для эффективности реализации требования, ограничивающего максимальный срок действия пароля. Далее рассмотрим атаки на парольную аутентификацию подробнее.

## Полный перебор паролей.

Полный *перебор паролей* – прямой перебор всех возможных сочетаний допустимых в пароле символов. Поиск можно несколько ограничить, указав число символов в пароле, их тип (буквы, цифры, спецзнаки) или задав символы, с которых должен начаться поиск.

Поиск можно несколько ограничить, указав число символов в пароле, их тип (буквы, цифры, спецзнаки) или задав символы, с которых должен начаться поиск.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Алфавит | Мощность алфавита | Время взлома паролей из 3 символов (сек) | Время взлома паролей из 6 символов | Время взлома паролей из 8 символов | Время взлома паролей из 12 символов |
| Маленькие латинские буквы | 26 | 0.02 | 5 мин. | 58 час | 3000 лет |
| Маленькие латинские буквы и цифры | 36 | 0.04 | 36 мин. | 32 дня | 150 000 лет |
| Маленькие и большие латинские буквы, цифры | 62 | 0.02 | 15 час | 7 лет | 100 млн. лет |
| Маленькие и большие латинские буквы, цифры и спецсимволы | 95 | 1 | 8 дней | 193 год | Дольше, чем существует планета Земля |

Таблица1. Иллюстрация зависимости времени полного перебора паролей от их длины (при скорости перебора паролей 1 000 000 паролей/сек)(1).

На сегодня NIST рекомендует для наилучшей защиты использовать 12-символьный пароль в 80-бит энтропии, который должен генерироваться с помощью 95-символьного алфавита (набор ASCII). Примеры ниже иллюстрируют ошибки, приводящие к генерации слабых паролей. Каждый из приведенных паролей основан на простом шаблоне, из чего следует очень низкая энтропия, что позволяет очень быстро их угадывать.

1. Семьянов П. статья «Частые опросы о взломе паролей»
2. Пароль по умолчанию: «password», «default», «admin», «guest» и др. Список паролей по умолчанию широко распространен по интернету.
3. Словарные слова: «chameleon», «RedSox», «sandbags», «bunnyhop!», «IntenseCrabtree» и др., включая слова из не английских словарей.
4. Слова с добавленными числами: «password1», «deer2000», «ivan1234» и др. Могут быть очень быстро проверены.
5. Слова с простой заменой букв: «p@ssw0rd», «l33th4x0r», «g0ldf1sh» и др. Могут быть проверены автоматически с небольшими временными затратами.
6. Удвоенныеслова: «crabcrab», «stopstop», «treetree», «passpass» идр.
7. Последовательности знаков, соответствующие рядом стоящим клавишам на клавиатуре: «qwerty», «12345», «asdfgh», «fred» и др.
8. Числовые последовательности, основанные на хорошо известных наборах: 911, 314159... или 271828..., 112358... и другие.
9. Личные данные пользователя: «ivpetrov123», «1/1/1970», номер телефона, "%username%, номер ИНН, адрес и другие.

Атаки с помощью словаря Атака с помощью словаря (dictionaryattack) – атака на систему защиты, использующая метод полного перебора паролей, при котором перебираются все слова определенного вида и длины из словаря (слова в чистом виде или их зашифрованные образы). Атака основана на предположении, что в пароле используются существующие слова какого-либо языка либо их сочетания.

Различают два вида таких атак: «online» атаки, в которых атакующему для проверки корректности пароля необходимо взаимодействие с сервером. «Offline» атаки, когда атакующий может проверить все допустимые пароли, не нуждаясь в обратной связи с сервером. Вероятностная оценка успеха атак по словарю равна отношению количества взломанных паролей при атаке по словарю к общему числу попыток. Для полного перебора или перебора по словарю могут использоваться специальные ПО:

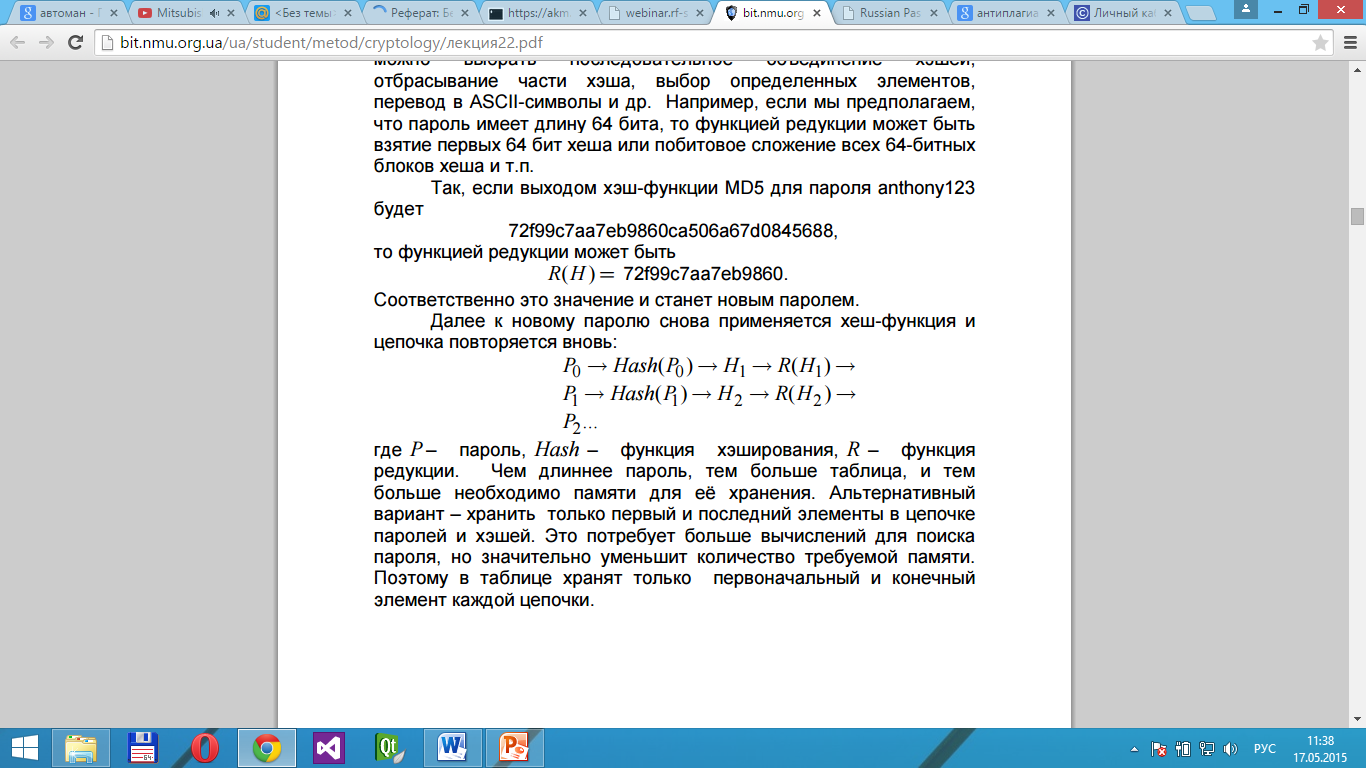
* PasswordsPro;
* MD5BFCPF;
* JohntheRipper.

Противодействия online атакам со словарем:

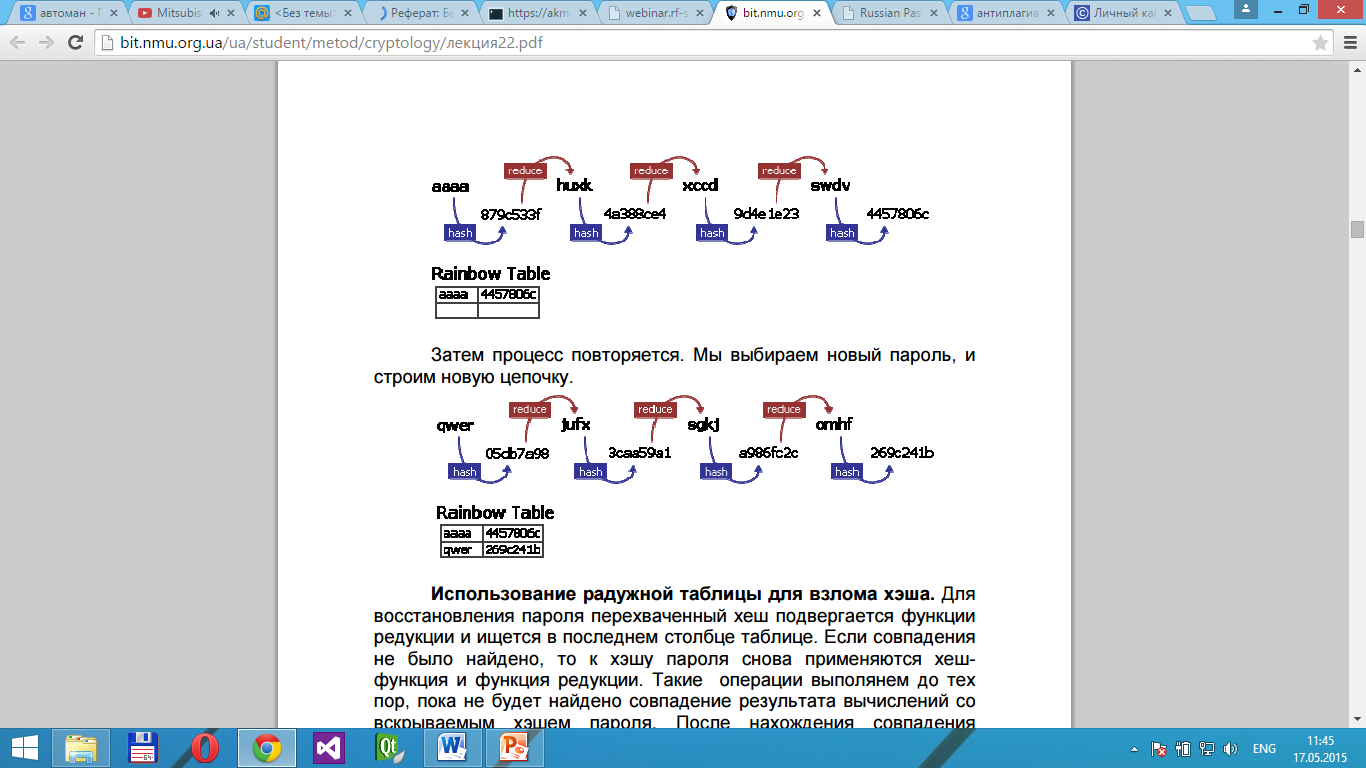
1. Задержка ответа (delayed response): для предоставленной пары логин/пароль сервер использует небольшую, специально сгенерированную задержку ответа (не чаще одного ответа в секунду);
2. Блокировка учетной записи (accountlocking) после нескольких неудачных попыток ввода пары логин/пароль (например, блокировка на час после пяти неправильных попыток ввода пароля). Первые два способа, в большинстве случаев, мешают проведению словарной атаке и взлому пароля за допустимое время.
3. Предполагается, что ввод верной комбинации логин/пароль производится реальным человеком, а атака по словарю – автоматической программой. Поэтому попытка ввода правильного пароля должна быть «вычислительно простой» для человека, и «вычислительно сложной» для машин. Чтобы сервер мог отличить человека от бота, используют обратный тест Тьюринга, другое название которого CAPTCНA, например, тест с использованием изображений перед вводом логин/пароля.
   * 1. **Атаки с помощью радужных таблиц.**

Атаки на пароли с помощью радужных таблиц (англ. rainbowtable) предложены Филиппом Окслином в 2003 году и обусловили существенное увеличение скорости подбора паролей. По сути, технология радужных таблиц представляет собой один из вариантов компромисса «время-память» и позволяет сократить перебор вариантов, например, паролей за счет использования больших предвычисленных таблиц. В таком случае атака состоит из двух этапов — трудоемкого построения таблиц, которое может быть выполнено заранее на мощном оборудовании, и быстрой атаки на хэш-образ пароля, которая может занимать несколько секунд или минут на обычном персональном компьютере. Создание таблицы. Принцип генерации радужной таблицы следующий: каждая цепочка начинается со случайного возможного пароля, который подвергается действию хеш-функции и функции редукции. Функция редукции – это функция, преобразующая выход хеш-функции в некий возможный пароль. В качестве такой функций можно выбрать последовательное объединение хешей, отбрасывание части хэша, выбор определенных элементов, перевод в ASCII-символы и др. Например, если мы предполагаем, что пароль имеет длину 64 бита, то функцией редукции может быть взятие первых 64 бит хеша или побитовое сложение всех 64-битных блоков хеша и т.п.

Так, пусть выходом хэш-функции для пароля будет *«72 F9 9С 7A A7 EB 98 60 CA 50 6A 67 D0 84 56 88»*, то функцией редукции может быть R(H) = *«72 F9 9C 7A A7 EB 98 60»*. Соответственно это значение и станет новым паролем. Далее к новому паролю снова применяется хеш-функция, и цепочка повторяется вновь:



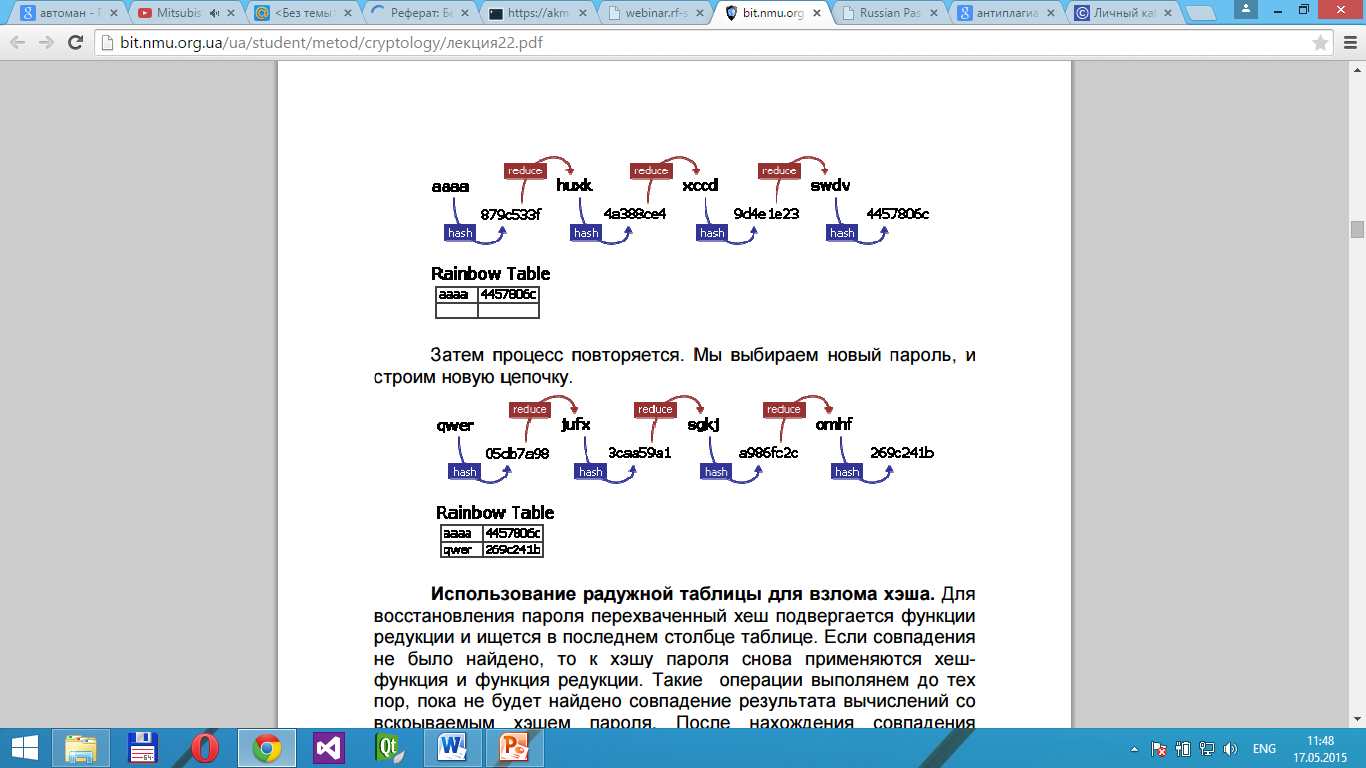
*P*– пароль, *Hash* – функция хеширования, *R* – функция редукции. Чем длиннее пароль, тембольше таблица, и тем больше необходимо памяти для её хранения. Альтернативный вариант – хранить только первый и последний элементы в цепочке паролей ихеш значений. Это потребует больше вычислений для поиска пароля, но значительно уменьшит количество требуемой памяти. Поэтому в таблице хранят только первоначальный и конечный элемент каждой цепочки.



**Rainbow Table**

|  |  |
| --- | --- |
| *aaaa* | *4457806C* |
|  |  |

Затем процесс повторяется. Мы выбираем новый пароль, и строим новую цепочку.



**Rainbow Table**

|  |  |
| --- | --- |
| *aaaa* | *4457806C* |
| *qwar* | *269C241B* |

Для усложнения подбора пароля и предотвращение атак с помощью радужных таблиц перед хешированием к паролю добавляют соль. Солью (salt) называется (псевдо) случайная битовая строка *salt*, добавляемая к аргументу password (паролю) функции хешированияℎ для рандомизации хеширования одинаковых сообщений. Соль передается вместе с вычисленным хеш значением. Соль удлиняет пароль, а это значительно осложняет восстановление исходных паролей с помощью предварительно построенных радужных таблиц. Ведь в этом случае атакующему необходимо не только сгенерировать таблицы с потенциальными паролями, но сгенерировать их с разным значением соли. Благодаря этому временная сложность атак по словарю или радужным таблицам увеличивается (напомним, что, однако, соль не защищает от полного перебора каждого пароля в отдельности). Например, пусть украден хешированный пароль пользователя, который является одним из 200 000 английских слов из словаря. Система использует 32-х битную соль. Добавление соли делает бессмысленными все посчитанные хеши паролей в радужных таблицах и последние становятся бесполезными. Злоумышленник теперь должен вычислять хеш каждого пароля с каждым из 32 2 4 294 967 296 = 2 32 (4 294 967 296) возможных вариантов соли до тех пор, пока не будет получено совпадение. Общее число возможных комбинаций хеш значений и соли: 32 14 2 200 000 000 858 993 459 10 – порядка 900 триллионов (вместо 200 000).

Для каждого пароля должна использоваться уникальная соль. Кроме того введение соли снижает вероятность парольных коллизий. Поэтому соль нерекомендуется повторять. Как пример, рассмотрим организацию парольной защиты в операционной системе Windows NT. Перед тем как получить доступ к ресурсам системы, пользователь должен пройти процедуру входа в систему, при этом подсистема безопасности должна распознать его по имени и проверить подлинность запроса по паролю. Пароль пользователя хранится в базе данных в двух вариантах – в виде, необходимом для проведения аутентификации между компьютерами, работающими под управлением ОС Windows 95. Для формирования NT hash используется алгоритм MD4. Для формирования Lanmanager hash все алфавитные символы пароля приводятся к верхнему регистру, каждая из двух половин 14- байтового символьного пароля обрабатывается независимо от другой. Если длина пароля меньше 14 символов, то вторая половина забивается нулями. На основании двух 7-байтовых половин пароля формируется ключ для шифрования DES некоторого 64-битного числа. В результате получаются две половины 16-байтового хешированного пароля.

Количественная оценка пароля. Пусть A – мощность алфавита паролей (число символов, которые можно использовать для составления пароля), L−длина пароля, тогда

*S =AL* – число всех возможных паролей длины L (пространство атаки);

*v* – скорость пароля подбора злоумышленником;

*T*– максимальный срок действия пароля.

Вероятность подбора пароля равна:

*P = =*

## Двухфакторная аутентификация. Преимущества и недостатки.

Многофакторная аутентификация имеет преимущества перед системами с однофакторной аутентификацией, так как системы с однофакторной аутентификацией не способны обеспечить высокую степень безопасности в современных системах. Для решения такой задачи многофакторная аутентификациядля того, чтобы пользователю получить доступ к ресурсам, требует от пользователя предоставить несколько факторов, по которым можно будет определить, что это именно тот человек, за которого он себя выдает. Частным случаем многофакторной аутентификации служит – *двухфакторная* аутентификация.

Наиболее часто распространённой вариацией использования аутентификации по двум фактором является сочетание смарт-карты и пароля пользователя. Смарт-карта является первым фактором, чем «владеет» пользователь, а пароль является вторым фактором аутентификации, то, что «знает» пользователь. Такой подход имеет ряд преимуществ:

* 1. повышается уровень защиты системы, за счёт введения дополнительного фактора;
  2. описанные ранее методы атаки на парольную аутентификацию, не могут применяться напрямую;
  3. снижение рисков, связанных с использованием слабых паролей:
  4. относительная низкая стоимость на рынке аппаратных идентификаторов, по сравнению средствами биометрической идентификации.

Однако, не существуют систем, которые были бы лишены недостатков. Двухфакторная аутентификация на основе смарт-карты и пароля, так же имеет недостатки. К ним можно отнести:

1. создаются дополнительные неудобства для пользователя;
2. возможны потери смарт-карт;
3. усложняется реализация и поддержка такой системы.

Двухфакторная аутентификация может быть реализована с использованием того, что «есть» у пользователя. Сюда можно отнести биометрические параметры человека: отпечатки пальцев, голос, сетчатка глаза. Однако, использование биометрических средств аутентификации так же имеет ряд недостатков. К ним можно отнести:

1. изменение биометрических идентификаторов, например, после болезни или несчастного случая;
2. существенно дороже на рынке аппаратные средства определения биометрических параметров человека, например, по сравнению со смарт-картами.
3. сложность реализации и поддержка таких систем.

Неоспоримым преимуществом, является уникальность биометрических параметров у каждого человека, из-за чего повышается безопасность аутентификации.

Таким образом, для решения поставленной задачи, было выбрана двухфакторная аутентификации с помощью смарт-карты и пароля.

## Криптографические протоколы аутентификации.

## Основные понятия.

*Протокол* – это последовательность шагов, которые предпринимают две или большее количество сторон для решения определённой задачи.Следует обратить внимание на то, что все шаги предпринимаются в порядке строгой очередности и ни один из них не может быть сделан прежде, чем закончится предыдущий.

*Криптографическим протоколом* называется протокол, в основе которого лежит криптографические алгоритмы. Основная цель таких протоколов является сохранение информации втайне от злоумышленников.

Криптографический протокол характеризуется следующими характеристиками:

* вычислительная эффективность – количество операций, необходимых для выполнения протокола;
* коммуникационная эффективность – данное свойство отражает количество сообщений и их длину, необходимую для осуществления аутентификации;
* наличие третьей стороны – примером третьей стороны может служить доверенный сервер распределения симметричных ключей или сервер, реализующий дерево сертификатов открытых ключей;
* основа гарантий безопасности – примером могут служить протоколы, обладающие свойством доказательства с нулевым знанием;
* хранения секрета – имеется в виду способ хранения критичной ключевой информации.

Функции криптографических протоколов:

* аутентификация источника данных;
* аутентификация сторон;
* обеспечение конфиденциальности данных;
* обеспечение целостности данных;
* разграничение доступа к информации.

Классификация криптографических протоколов:

* протоколы шифрования и дешифрования;
* протоколы аутентификации и идентификации;
* протоколы обмена ключами шифрования;
* протоколы электронно-цифровой подписи.

В основе *протоколов шифрования и дешифрования* лежат симметричные и ассиметричные алгоритмы шифрования. Например: российский алгоритм симметричного шифрования *ГОСТ 28147-89*. Алгоритм на вход принимает открытый текст и ключ шифрования, на выходе алгоритм выдает зашифрованный переданным ключом информацию. Процесс дешифрования является обратным процессу шифрования. На вход алгоритма подаётся зашифрованный тест и ключ шифрования, по которому было зашифрована информация. В итоге алгоритм на выходе выдает дешифрованную информацию.

## 2. 2. Аутентификация клиента методом «запрос - ответ».

Идея протоколов аутентификации «запрос - ответ» (challenge-response) состоит в том, что одна сторона (претендент) доказывает свою идентичность другой стороне (проверяющему), демонстрируя ей знание секрета (в некоторых протоколах секрет известен проверяющему и используется для проверки ответа, в других - вообще нет необходимости, чтобы секрет был известен проверяющему). Претендент должен ответить на запрос, меняющийся во времени, причем ответ должен зависеть и от его секрета и от запроса. Обычно запрос – это число, выбираемое одной стороной в начале протокола. Для противодействия атакам на протокол методами повтора сеанса используют параметры, меняющиеся во времени. Это гарантирует уникальность и актуальность каждого сеанса протокола. В криптографических протоколах широко используется понятие свежести (freshness) той или иной величины. Оно означает, что значение величины было сгенерировано в начале или входе выполнения текущего сеанса протокола.

В качестве параметров, обеспечивающих эту свежесть, используют три типа параметров:

* Числовые последовательности. Например, последовательность номеров сеансов протокола с увеличением на единицу в каждом последующем сеансе. Недостатком этого способа – необходимость ведения участниками счетчиков последовательностей чисел.
* Случайные числа. Легко генерируются всеми сторонами протокола с использованием генератора случайных/псевдослучайных чисел. Обычно случайные числа обозначают символом r с соответствующими индексами.
* Метки времени (timestamps). Сторона, генерирующая сообщение, получает метку времени со своих локальных часов (системного таймера) и c помощью криптографии привязывает ее к сообщению. Получатель берет текущее время со своих локальных часов и сравнивает с величиной метки времени, полученной от партнера.

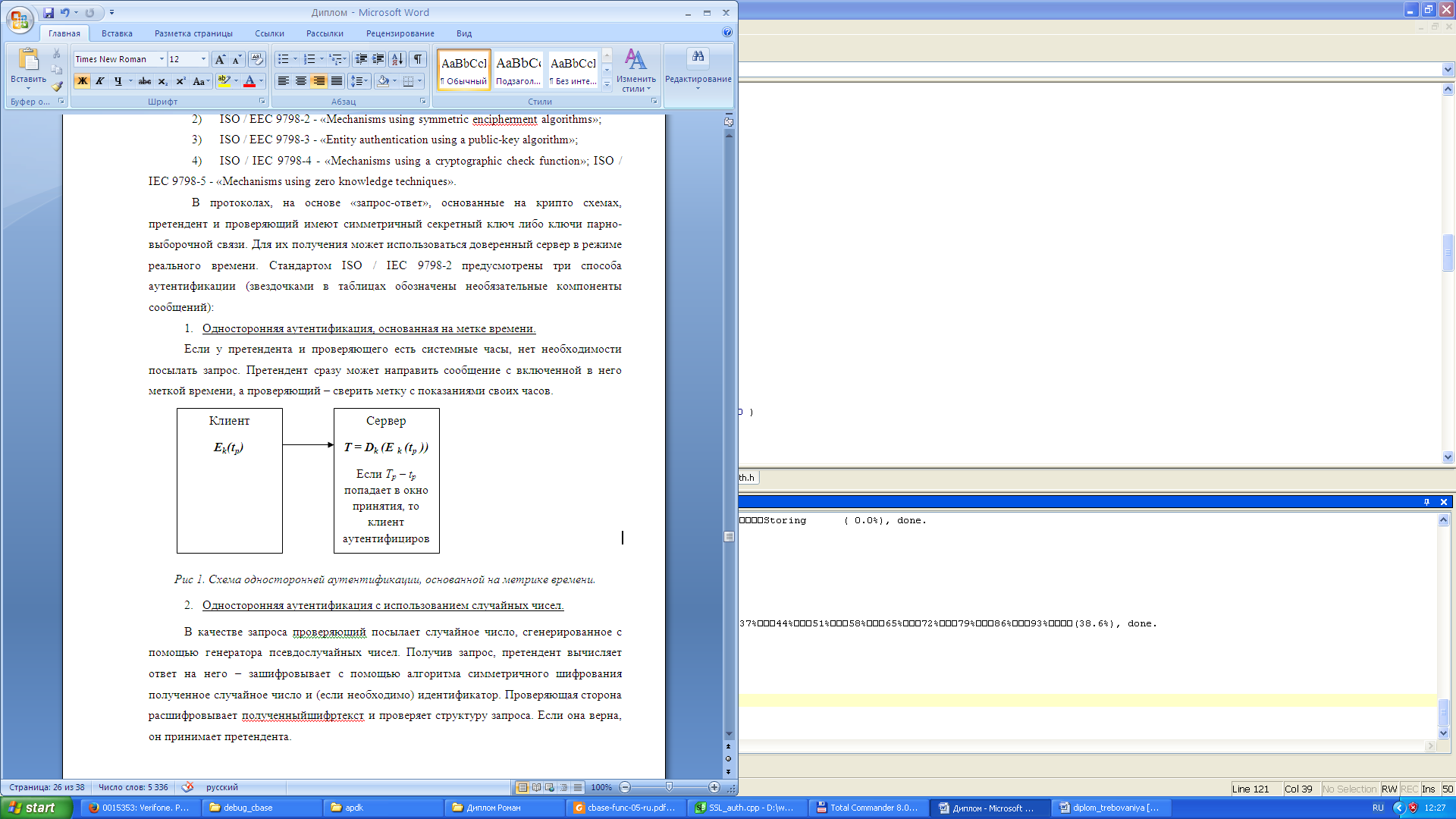
Для использования этого метода необходимо, чтобы:

1. разница между метками времени отправителя и получателя укладывалась в определенный интервал времени – окно принятия (accept ance window);
2. ранее не поступали сообщения с идентичной меткой времени от того же отправителя;
3. часы были синхронизированы. Преимущество третьего метода – не надо поддерживать внутренний счетчик чисел. Системный таймер есть практически на любой вычислительной платформе. Основным международным стандартом по криптографическим протоколам аутентификации является стандарт Международной организации по стандартизации и Международной электротехнической комиссии ISO / IEC 9798, состоящий из пяти частей:
4. ISO / IEC 9798-1 - «General Model»;
5. ISO / EEC 9798-2 - «Mechanisms using symmetric encipherment algorithms»;
6. ISO / EEC 9798-3 - «Entity authentication using a public-key algorithm»;
7. ISO / IEC 9798-4 - «Mechanisms using a cryptographic check function»; ISO / IEC 9798-5 - «Mechanisms using zero knowledge techniques».

В протоколах, на основе «запрос-ответ», основанные на крипто схемах, претендент и проверяющий имеют симметричный секретный ключ либо ключи парно-выборочной связи. Для их получения может использоваться доверенный сервер в режиме реального времени. Стандартом ISO / IEC 9798-2 предусмотрены три способа аутентификации (звездочками в таблицах обозначены необязательные компоненты сообщений):

1. Односторонняя аутентификация, основанная на метке времени.

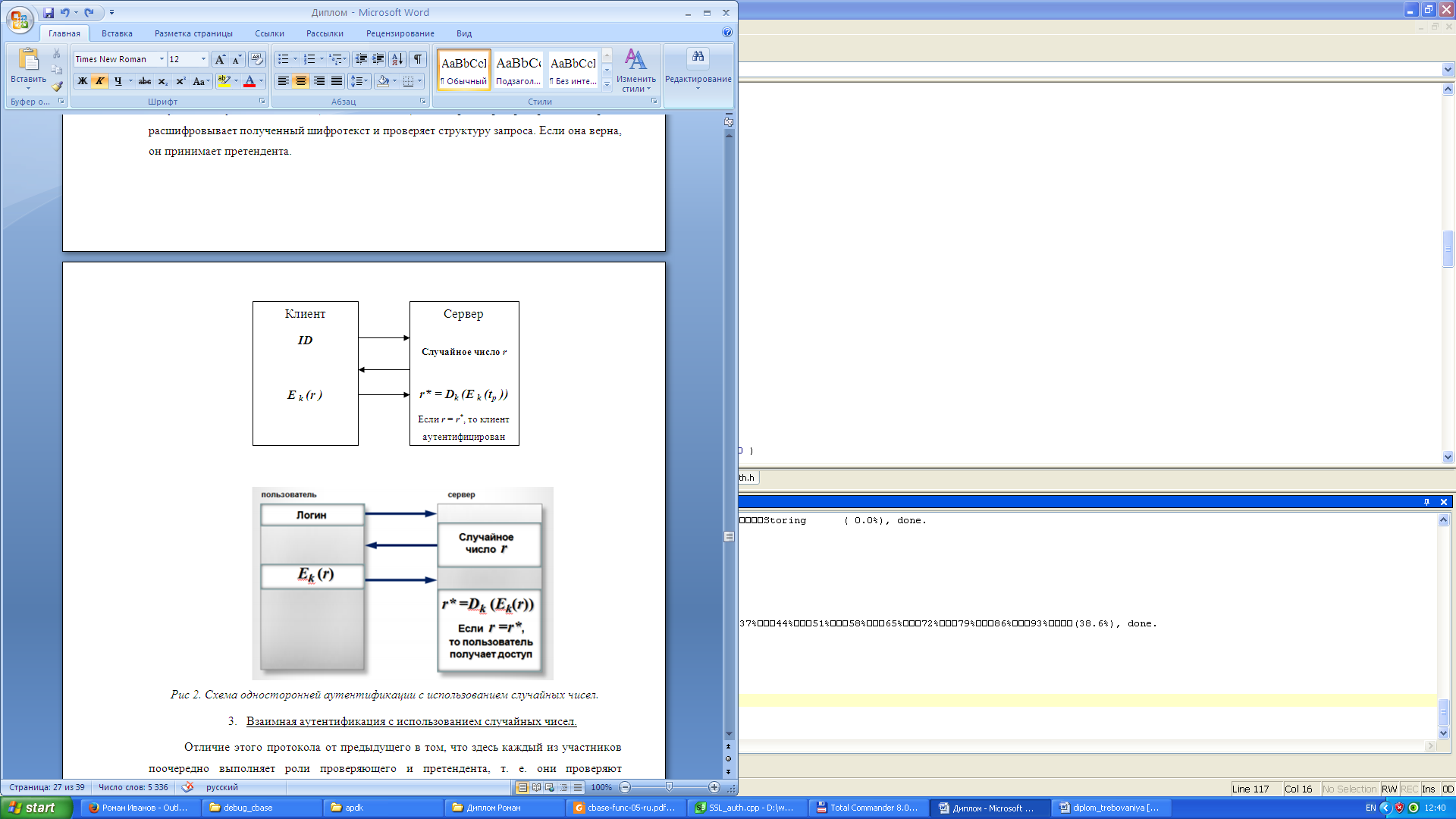
Если у претендента и проверяющего есть системные часы, нет необходимости посылать запрос. Претендент сразу может направить сообщение с включенной в него меткой времени, а проверяющий – сверить метку с показаниями своих часов.

****

*Рис 1. Схема односторонней аутентификации, основанной на метрике времени.*

1. Односторонняя аутентификация с использованием случайных чисел.

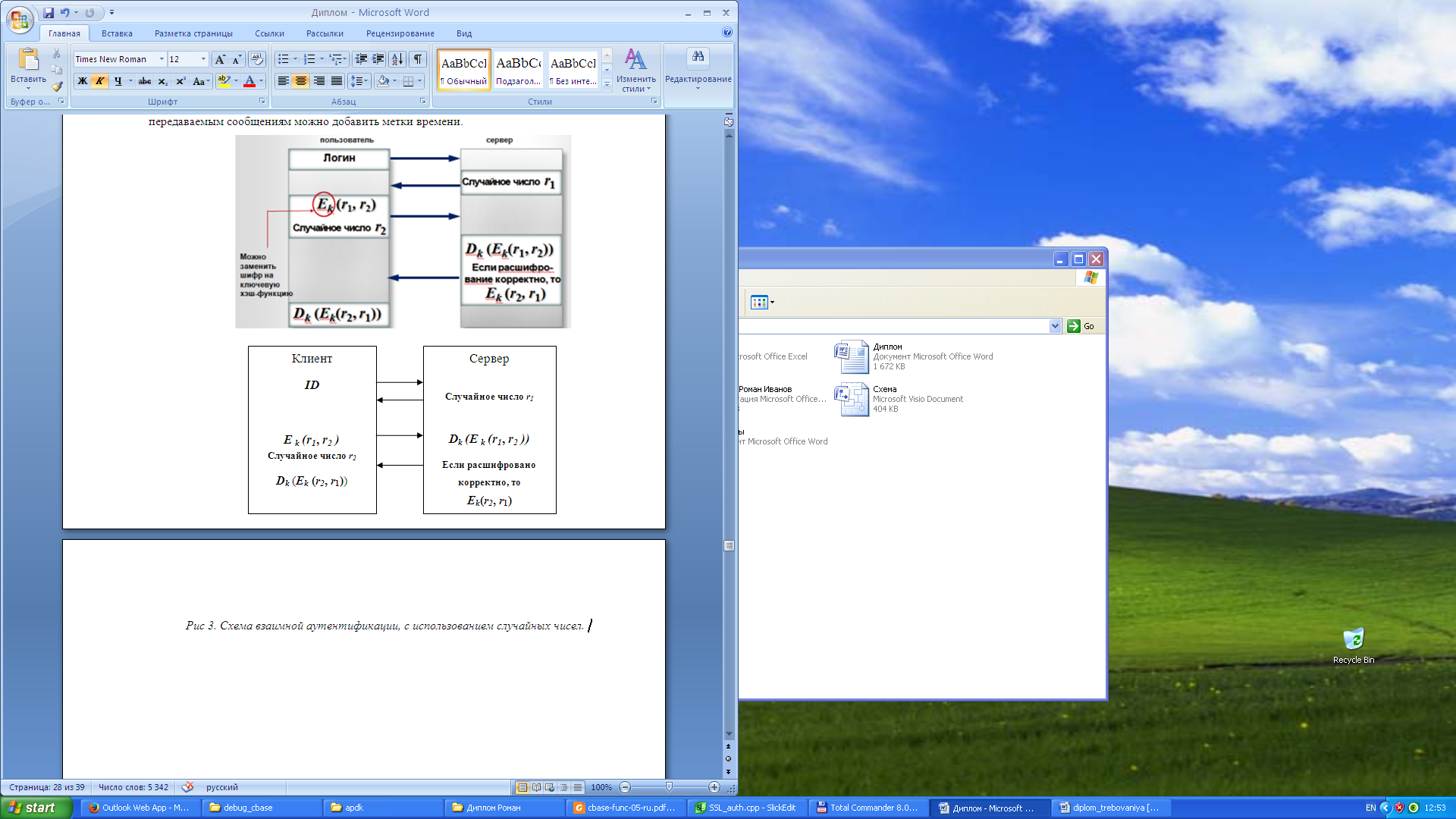
В качестве запроса проверяющий посылает случайное число, сгенерированное с помощью генератора псевдослучайных чисел. Получив запрос, претендент вычисляет ответ на него – зашифровывает с помощью алгоритма симметричного шифрования полученное случайное число и (если необходимо) идентификатор. Проверяющая сторона расшифровывает полученный шифротекст и проверяет структуру запроса. Если она верна, он принимает претендента.



*Рис 2. Схема односторонней аутентификации с использованием случайных чисел.*

1. Взаимная аутентификация с использованием случайных чисел.

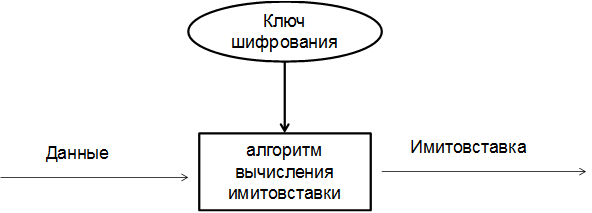
Отличие этого протокола от предыдущего в том, что здесь каждый из участников поочередно выполняет роли проверяющего и претендента, т. е. они проверяют аутентичность друг друга. Протокол взаимной аутентификации – это по сути два протокола односторонней аутентификации, «упакованные» в три пересылки сообщения. Подобного рода протоколы в силу их симметричности называются протоколами рукопожатия. Этот протокол допускает замену шифра на хеш-функцию с ключом, как указано в стандарте ISO / IEC 9798-4. Для повышения стойкости протокола к передаваемым сообщениям можно добавить метки времени.

****

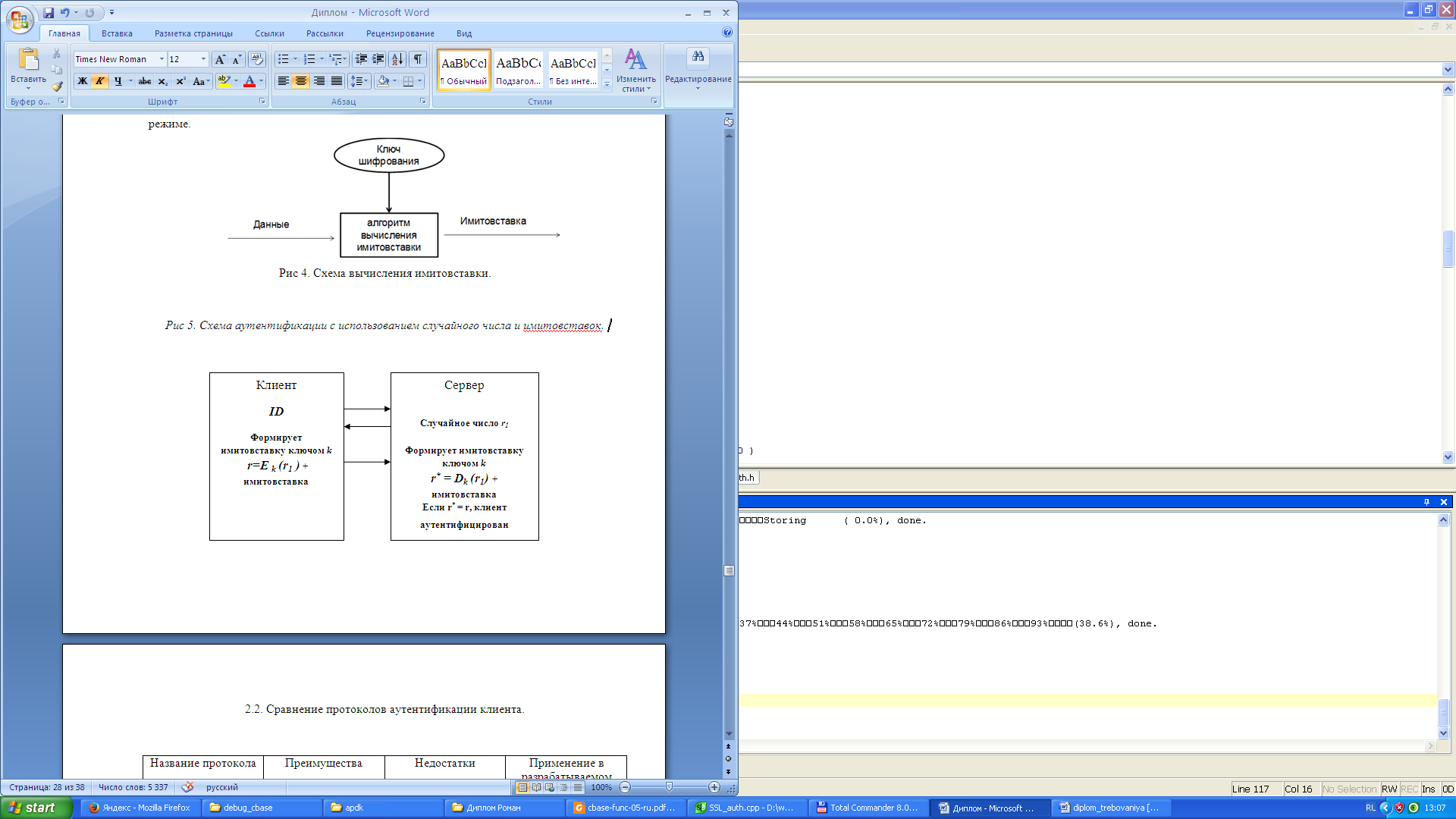
*Рис 3. Схема взаимной аутентификации, с использованием случайных чисел.*

1. Аутентификации с использованием случайных чисел и имитовставок.

*Имитовставка* – значение, рассчитанное по входным данным с помощью криптографического алгоритма с использованием секретного элемента (ключа), известного только отправителю и получателю. Имитовставка необходима для аутентификации сообщения, получаемое от клиента.



*Рис. 4. Схема вычисления имитовставки.*

****

*Рис. 5. Схема аутентификации с использованием случайного числа и имитовставок.*

Клиент, после соединения с сервером, отправляет ID (идентификатор), после чего сервер генерирует случайное число и отправляет клиенту. Клиент на заранее известном ключе вычисляет имитовставку и шифрует полученное от сервера случайное число на этом же ключе, после отправляет серверу. Сервер, зная ID клиента, использует заранее известный ключ и расшифрует сообщение. После исходное случайное число и добавленное к нему имитовставка соответствуют, тому, что прислал клиент, то аутентификация прошла успешно.

2.2. Сравнение протоколов аутентификации клиента.

|  |  |  |  |
| --- | --- | --- | --- |
| Название протокола | Преимущества | Недостатки | Применение в разрабатываемом протоколе |
| Односторонняя аутентификация, основанная на метке времени |  |  | нет |
| Односторонняя аутентификация с использованием случайных чисел |  |  | нет |
| Взаимная аутентификация с использованием случайных чисел |  |  | нет |
| Аутентификация с использованием имитовставки |  |  | да |

## 2.3. Протокол обмена сессионным ключом.

Для передачи секретной информации по открытым каналам связи, для защиты которой используется симметричное шифрование, абонентам необходимо иметь общий секретный ключ. Постоянно использование одного и того же ключа при каждомсеансе связи между абонентами, злоумышленнику позволяет накопить богатый для крипто анализа. Поэтому в целях повышения безопасности обмена секретной информации широко используются сессионные ключи. 

**Сессионный ключ** – это ключ, который используется абонентами в рамках одного сеанса (сессии, раунда) общения. Использование сеансовых ключей позволяет так же ограничить размер ущерба при компрометации ключа.

Возможны следующие разновидности протоколов обмена ключами

* ключ вырабатывается одним из абонентов и высылается второму для последующего обмена информацией;
* совместная выработка ключа абонентами;
* ключ вырабатывается и предоставляется абонентам третьей стороной (доверенным центром).

Наиболее оптимальным и безопасным методом обмена сессионным ключом, является совместная выработка сессионного ключа. Одним из распространённых алгоритмов обмена сессионным ключом является протокол Диффи-Хеллмана.

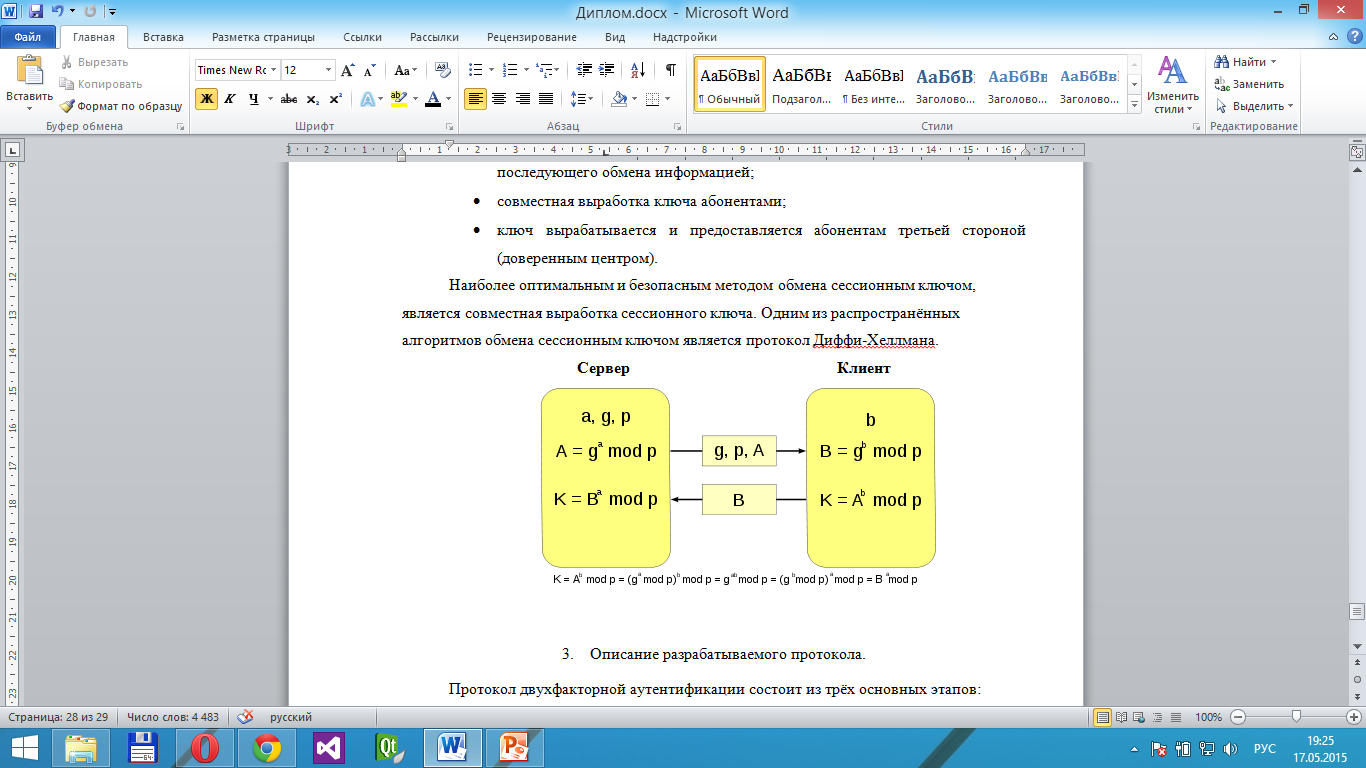


Рис. 6. Обмен ключами по протоколу Диффи – Хеллмана.

На первом этапе сервер генерирует клиенту параметры *p*, *g* и *a*, где *p*и *g*общие, не секретные параметры алгоритма, *a*–секретный параметр сервера. Затем вычисляет публичный ключ на основе секретного параметра *a* по формуле *A = gamodp*. Параметры *p*, *g* и *a* передаются клиенту.

На втором этапе клиент генерирует секретный параметр*b*и, на основе полученный от сервера параметров *p* и *g*,вычисляет открытый ключ клиента по формуле *B = gb mod p* и передает серверу *B*.

На третьем этапе сервер и клиент вычисляют, по имеющим параметрам, общий секретный сессионный ключ по формулам *K = Ba mod p* и *K = Ab mod p* соответственно.

ПРАКТИЧЕСКИЙ РАЗДЕЛ

## Описание разрабатываемого криптографического протокола.

* 1. **Общие сведения.**

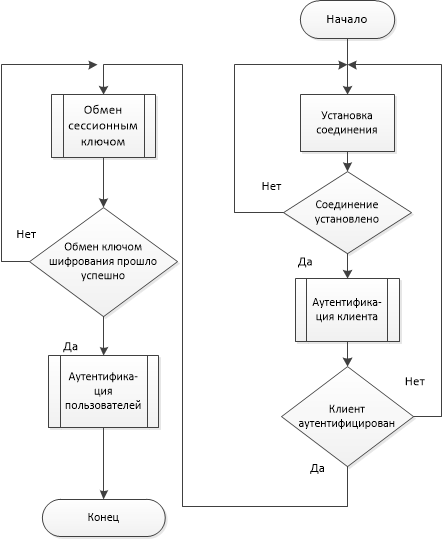
При разработке протокола необходимо решить следующие основные задачи:

1. Выработать механизм проверки подлинности клиента, инициирующее соединение с сервером.
2. Обеспечить безопасность пользовательских данных, при передаче по каналу связи.

Для решения первой задачи, наиболее подходящим алгоритмом, является аутентификация клиента на основе случайных чисел и имитовставок. Для решения второй задачи было выбрано симметричное шифрование на основе российского стандарта ГОСТ 28147-89. Применение симметричного шифрования, требует необходимость выработки сессионного ключа. Эту задачу можно решить алгоритмом Диффи-Хеллмана.

Таким образом, протокол условно делиться на три этапа:

* + 1. Проверка подлинности клиента, инициирующее соединение
    2. Обмен сессионным ключом симметричного шифрования
    3. Безопасная передача пользовательских данных, предназначенных для двухфакторной аутентификация, по открытому каналу связи.



*Рис. 7. Алгоритм криптографического протокола.*

На рис.7, представлен алгоритм криптографического протокола. На первом шаге устанавливается соединение клиента с сервером. Когда соединение будет установлено, начинается процесс проверки подлинности клиента. Если клиент аутентифицирован успешно, то переход к обмену сессионным ключом симметричного шифрования по протоколу Диффи-Хеллмана. После обмена сессионным ключом можно перейти к процессу аутентификации пользователей, по безопасному каналу связи.

* 1. **Описание первого этапа протокола.**

Из рассмотренных выше протоколов проверки подлинности клиента, наиболее подходящим для данного случая, является аутентификация с использованием случайных чисел и имитовставок. Аутентификация односторонняя, так как только серверу необходимо проверить подлинность клиента. Клиент и сервер заранее имеют секретный общий ключ. Сервер отправляет клиенту случайное число *r*1. Клиент имеющимся у него ключом шифрования, шифрует *r*1 в режиме имитовставки, затем формирует сообщение и отправляет серверу.



*Рис.8. Алгоритм проверки подлинности клиента сервером.*

Сервер в БД ищет ID (идентификатор) клиента, если идентификатор клиента найден, то по соответствующему данному идентификатора ключу, шифрует случайное число *r1* в режиме имитовставки и сравнивает с данными, полученными от клиента. Если результат совпадает, с полученным сообщением, то клиент проходит проверку подлинности на сервере.

* 1. **Описание второго этапа протокола.**

Второй этап протокола представляет с собой алгоритм обмена сессионным ключом для симметричного шифрования.



*Рис.9. Алгоритм обмена сессионным ключом со стороны сервера.*

Сервер генерирует простые числа *p* и *g*, которые являются общими и не секретными параметрами алгоритма Диффи-Хеллмана и передает их клиенту. Для обеспечения контроля целостности передаваемых данных, клиент, после получения сообщения, отправляет серверу, вычисленные хеш значения этих параметров. После успешного обмена параметрами *p* и *g*, сервер генерирует секретное число *a*, вычисляет публичный ключ сервера по формуле *A = g a mod p* и передает его клиенту. Так же, как и с первыми параметрами, в целях обеспечения целостности передаваемых данных, клиент, после получения от сервера публичного ключа, передает обратно его хеш значение. После обмена публичным ключом сервера, следует обменяться публичным ключом клиента.



*Рис.10. Блок-схема алгоритма обмена сессионным ключом со стороны клиента.*

Для этого клиент, так же, генерирует секретное число *b*, вычисляет *B* по формуле *B=gbmod p* и отправляет результат серверу. Сервер вычисляет хеш значение от *B* и отправит клиенту. После успешного обмен параметрами Диффи-Хеллмана и пбличными ключами, клиента и сервер имеют всю информацию для вычисления сессионного ключа. Оба участника протокола, далее вычисляют сессионный ключ симметричного шифрования.

Таким образом, работает второй этап данного протокола, который предназначен для обмена ключами.

* 1. **Описание третьего этапа протокола.**

Третий этап протокола, представляет с собой безопасный обмен пользовательскими данными, для двухфакторной аутентификации. Сюда относиться идентификатор смарт-карты пользователя и его персональный пароль.



*Рис.11. Алгоритм аутентификации пользователей со стороны сервера.*

Клиент считывает, с помощью USB считывателя, идентификатор смарт-карты пользователя, зашифрует сообщение с сессионным ключом шифрования, который был выработан, на втором этапе проткала, и отправляет серверу. Сервер расшифрует сообщение сессионным ключом и проверяет в базе данных идентификатор клиента. Ели идентификатор был найден, то формирует ответ клиенту о необходимости требования пароля от пользователя, шифрует сообщение и отправляет. Клиент, получив ответ от сервера, предлагает пользователю ввести пароль.



*Рис.12. Алгоритм аутентификации пользователей со стороны клиента.*

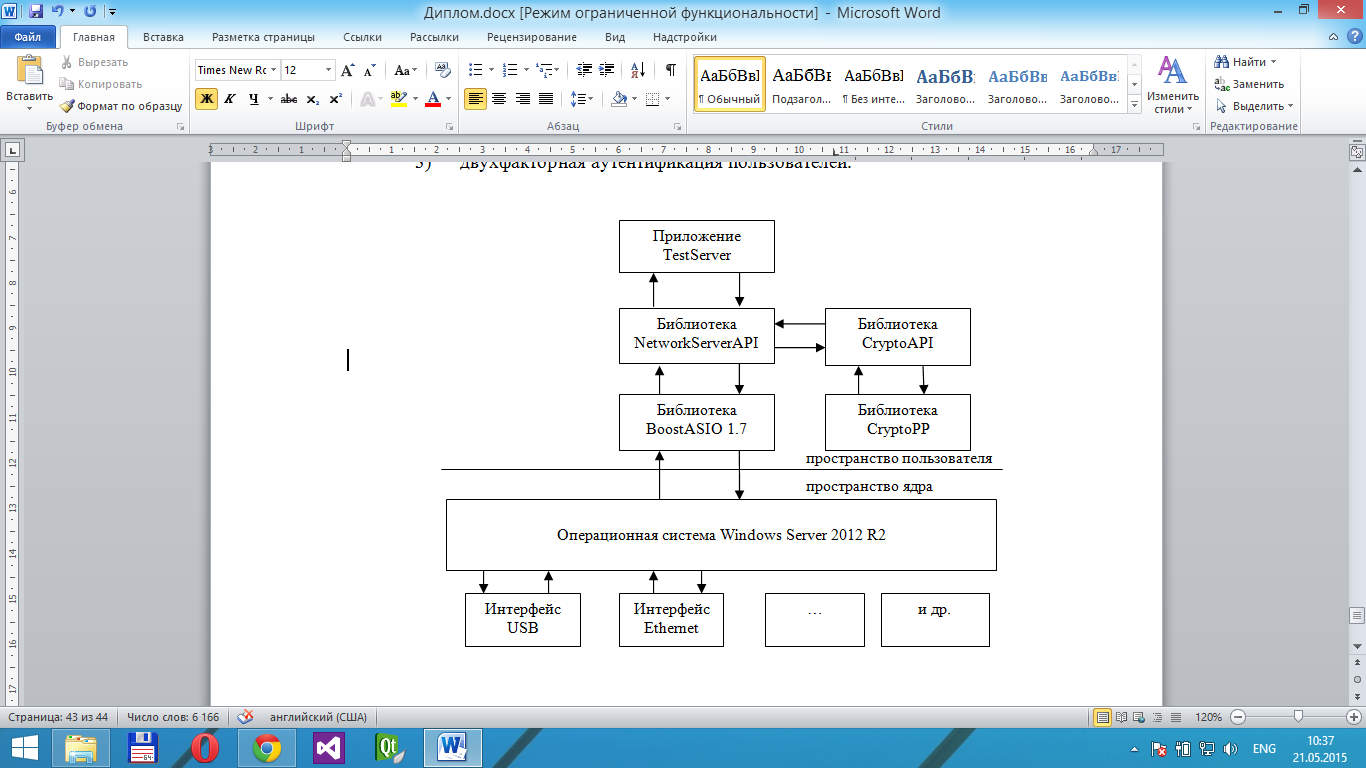
После ввод пароля, клиент вычисляет его хеш значение, шифрует сессионным ключом и отправляет серверу. Сервер, получив сообщение, дешифрует его, проверяет в БД на соответствие идентификатора клиента и хеш значения пароля. Если идентификатор соответствует полученному хеш значению от пароля, то сервер формирует ответ об успешной аутентификации пользователя, иначе аутентификация пользователя не было пройдено.

Таким образом, реализуемый в рамках данной задачи протокол состоит из трёх основных этапов. Выполнения всех трёх этапов позволяет решить все поставленные задачи.

**Реализация криптографического протокола.**

* 1. **Архитектура.**

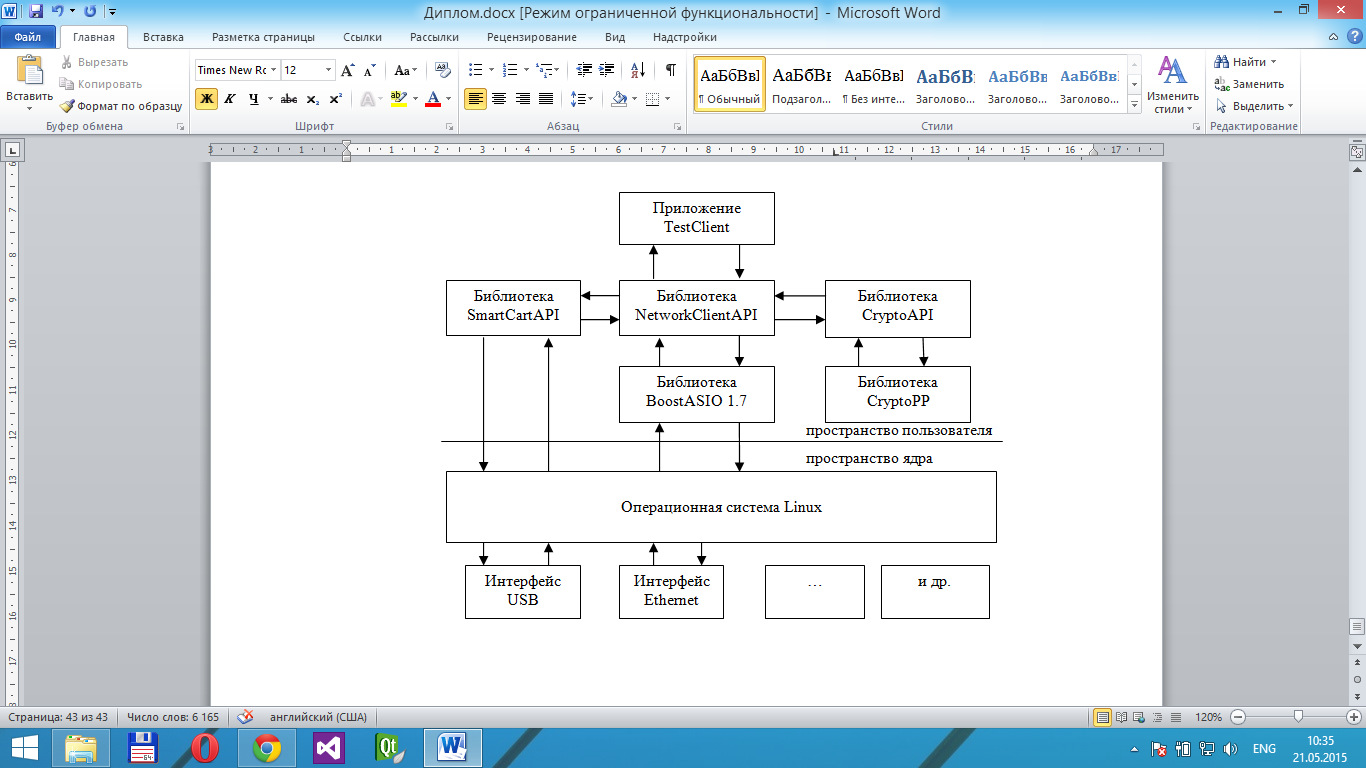
Согласно требованиям технического задания, протокол со стороны сервера должен быть реализован на платформе Windows Server 2012 R2, результатом работ должна быть библиотека, предоставляющий интерфейс доступа к функциям протолкла.



*Рис.13. Зависимости модулей протокола со стороны сервера.*

На *рис.13* представлена зависимость модулей протокола со стороны сервера. Основной библиотекой, где реализован интерфейс доступа к вышеописанному протоколу, является библиотека «NetworkServerAPI». Эта библиотека связано с «CryptoAPI», которая реализует криптографические алгоритмы, используемые в протоколе. «CryptoAPI» в своей реализации, применяет функции из кроссплатформенной библиотеки «CryptoPP», включающая в себе арифметические и логические операции над длинными числами (в данном случае длина до 256 бит). Сетевая часть зависит от функций, кроссплатформенной библиотеки BoostASIO, которая уже обращается к элементам ядра операционной системы Windows Server 2012 R2 через интерфейс WinAPI. Приложение «TestServer» предназначен для тестирования протокола со стороны сервера. Такая архитектура является гибким, имеет возможность расширения, благодаря разделению на независимых модулей, которые базируются на кроссплатформенных библиотеках.

Со стороны клиента модули протокола реализованы схожим образом и имеют такие же преимущества, как и у сервера.



*Рис.14. Зависимости модулей протокола со стороны клиента.*

На *рис.14*, представлены зависимости модулей протокола со стороны клиента. Основной библиотекой здесь является – «NetworkClientAPI», где реализован интерфейс доступа к протоколу со стороны клиента. Криптографические алгоритмы используются из «CryptoAPI», сетевой интерфейс предоставляет «BoostASIO», протокол доступа к считывателю смарт-карты реализовано в библиотеке «SmartCartAPI».

* 1. **Алгоритм симметричного шифрования ГОСТ 28147-89.**

ГОСТ 28147-89 является российским криптографическим стандартом симметричного шифрования. Алгоритм, описанный в данном стандарте, является блочным шифром, длина ключа которого составляет 256 бит, основан на сети Фейстеля. Шифрование происходит в 32 раунда, длина входного блока данных составляет 64 бит информации. Вектор инициализации алгоритма шифрования может иметь различные варианты, поэтому он должен предоставляться в отдельном файле. ГОСТ 28147-89 может работать в одном из четырех режимах:

1. режим простой замены;
2. гаммирование;
3. гаммирование с обратной связью;
4. режим выработки имитовставки.

Для реализации описанного выше протокола понадобиться два режима работы:

* режим простой замены;
* режим выработки имитовставки.

Для реализации алгоритма был выбран язык программирования Си. Имеется 4 интерфейсных функций:

1. инициализации (gost\_init);
2. шифрования в режиме простой замены (gostcrypt);
3. дешифрование в режиме простой замены (gostdecrypt);
4. выработка имитовставки (gostimito\_crypt);

Полное описание функций, реализованных на языке Си для алгоритма ГОСТ 28147-89, см. Приложение 1.

Функция *инициализация* принимает в качестве параметра путь к файлу с вектором инициализации в формате \*.bin. Загружает вектор инициализации и выполняет подготовительные этапы алгоритма.

Функция *шифрования в режиме простой замены*, в качестве параметров на вход принимают:

1. массив данных для шифрования размером 64 бит;
2. массив данных, для заполнения результата выполнения функции (зашифрованные данные);
3. массив размером 256 бит, содержащий ключ шифрования.

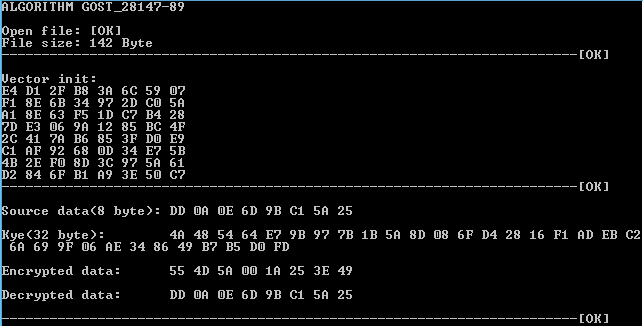
Функция *дешифрования в режиме простой замены* имеет параметры:

1. массив зашифрованных данных, размером 64 бит;
2. массив данных, для заполнения результата дешифрования, размером 64 бит;
3. массив размером 256 бит, который содержит ключ, по которому зашифрованы данные.

Функция *выработки имитовставки* принимает:

1. массив данных размером 64 бит;
2. массив для результата, размером 32 бит (вычисленная 32 битная имитовставка).

Рассмотрим результат работы алгоритма.



*Рис.15. Результат работы алгоритма ГОСТ 28147-89.*

На *рис.15* показан результат работы алгоритма ГОСТ 28147-89. Вначале, открывается файл размером 142 байта, содержащий вектор инициализации, считывается содержимое файла и производится подготовительный этап. Затем, продемонстрирована работа функции шифровании в режиме простой замены. 64 бита данных шифруются 256 битным ключом. В результате зашифрованные указанным ключом данные. Для проверки правильности работы функции выполняется обратный процесс шифрованию – дешифрование. В итоге дешифрованные данные совпадают исходными данными. Отсюда можно сделать вывод о правильности работы алгоритма.

* 1. **Алгоритм вычисления хеш-функция ГОСТ Р 11.34-2012.**

ГОСТ Р 11.34-2012 – российский криптографический стандарт, который определяет алгоритм вычисления хеш-функции. Алгоритм, описанный в этом стандарте, имеет в два режима:

* + 1. Режим вычисления хеш-функции, длина которой 256 бит;
    2. Режим вычисления хеш-функции, длина которой 512 бит.

Интерфейсные функции, реализующие данный криптографический стандарт, следующие:

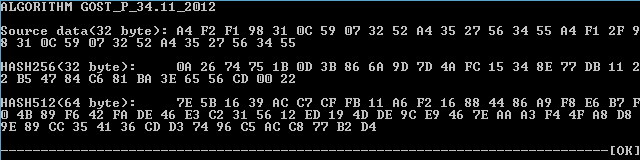
1. функция, вычисляющая хеш-функцию, значение которого 256 бит (hash\_256);
2. функция, вычисляющая хеш-функцию, значением 512 бит (hash\_512).

Полное описание функций см. Приложение 2.

Функция hash\_256, принимает в качестве параметра массив данных, от которых необходимо вычислить хеш значение и массив размером 256 бит для получения результата.

Функция hash\_512, так же принимает в качестве параметра массив данных, от которых необходимо вычислить хеш значение и массив размером 512 бит, куда будет записан результат.

Рассмотрим результат работы алгоритма вычисления хеш-функций.



*Рис.16. Результат работы алгоритма вычисления хеш-функций.*

На рис. 16 представлен результат работы алгоритма вычисления хеш-функций. Из исходных данных размером 32 байта, вычислено значение хеш-функции 256 бит и 512 бит.

* 1. **Алгоритм обмена сессионным ключом по протоколу Диффи – Хеллмна.**

Протокол обмена сессионным ключом Диффи-Хеллмна состоит из нескольких этапов (см. п. 2.3), в каждом из которых производятся арифметические и логические операции с очень длинными числами, так как длина ключа шифрования ГОСТ 28147-89 составляет 256 бит. Поэтому для реализации данного алгоритма использовано дополнительная библиотека с открытым исходным кодом, написанное на языке С++ - CryptoPP.

Библиотека поддерживает платформы:

* ОС Windows;
* ОС Linux.

Интерфейс доступа к реализации протокола обмена сессионным ключом реализует класс (diffy\_helman), содержащий следующие открытые методы (полное описание см. Приложение 3):

1. метод для генерации параметра *p* (get\_p);
2. метод для генерации параметра *g* (get\_g);
3. метод для расчёта публичного ключа (generate\_A);
4. метод для расчёта сессионного ключа (generate\_K).

Метод для *генерации параметра* *p*, в качестве параметра получает массив, в куда будет заполнен результат, после выполнения функции.

Метод *для генерации параметра* *g*, так же в качестве параметра получает массив, для получения результата функции.

Метод *для расчёта публичного ключа*, в качестве параметра получает публичный ключ клиента, если функцию использует сервер или публичный ключ сервера, если использующий – клиент.

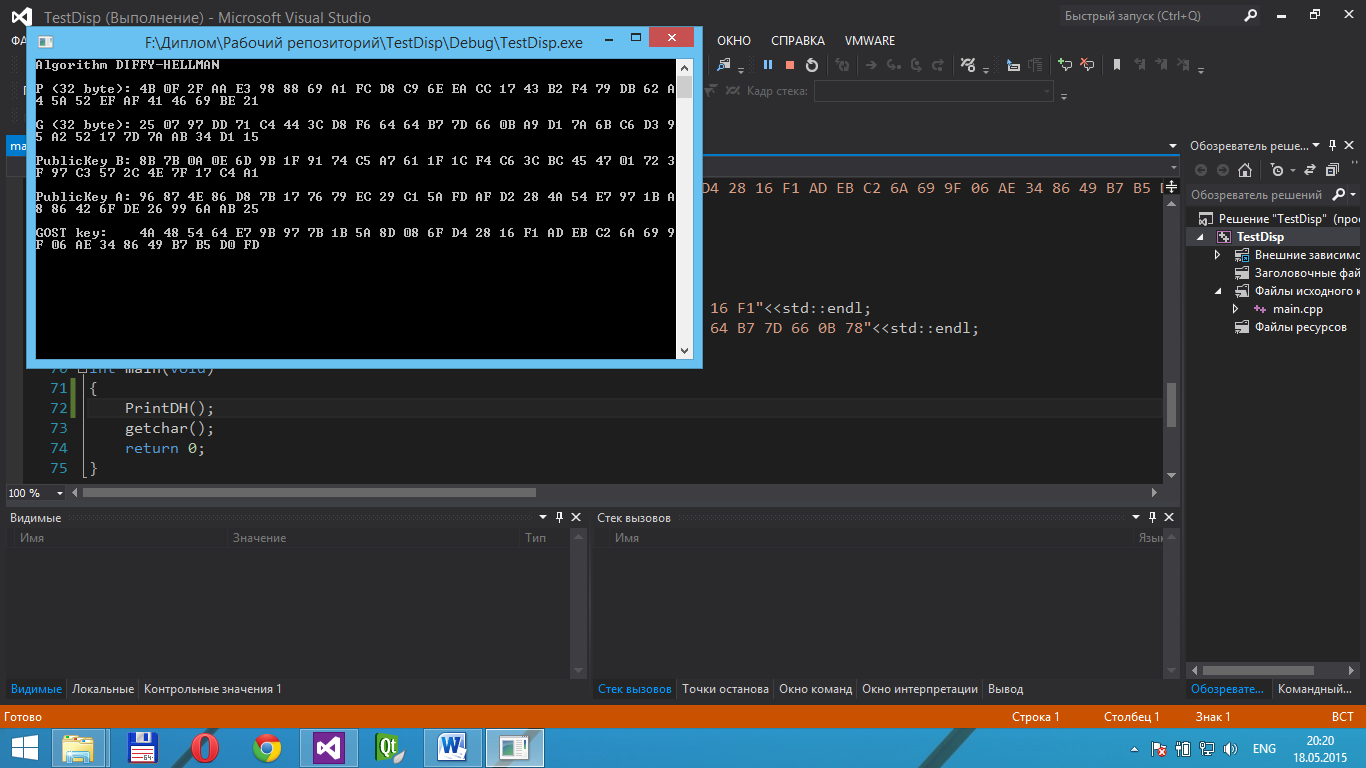
Метод для расчёта сессионного ключа, принимает два массива:

1. массив, который содержит публичный ключ;
2. массив для получения результата выполнения функции.

Таким образом, реализован интерфейс доступа к протоколу обмена сессионным ключом Диффи-Хелмана.

Теперь рассмотрим пример, демонстрирующий работу алгоритма. На рис. 14 продемонстрировано реализация алгоритма Диффи-Хеллмана.

Сгенерирован параметр *p* в 256 бит, параметр *g* в 256 бит. На основе этих параметров вычислен публичный ключ для сервера и публичный ключ для клиента. Затем, на основе публичных ключей клиента и сервера вычислен сессионный ключ.



*Рис.17. Результат работы алгоритм обмена сессионным ключом Диффи-Хеллмана.*

Таким образов, *рис.17* наглядно демонстрирует обмен сессионным ключом шифрования.

**Реализация протокола двухфакторной аутентификации пользователей на основе российских криптографических алгоритмов.**

**5.1. Реализация первого этапа протокола.**

Первый этап проткала, реализует аутентификацию клиента. Клиент инициирует соединение с сервером. После установки соединения начинается процесс аутентификации клиента. Протокол аутентификации реализован