УНИВЕРСИТЕТ ИТМО

С.В. Таранов

КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ И СТАНДАРТЫ. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

Санкт-Петербург
2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

С.В. Таранов

КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ И СТАНДАРТЫ. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

РЕКОМЕНДОВАНО К ИСПОЛЬЗОВАНИЮ В УНИВЕРСИТЕТЕ ИТМО
по направлению подготовки 10.04.01 Информационная безопасность
в качестве Учебно-методическое пособие для реализации основных профессиональных образовательных программ высшего образования магистратуры

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург
2022

Таранов С.В., КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ И СТАНДАРТЫ. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ– СПб: Университет ИТМО, 2022. – 37 с.

Рецензент(ы):
Комаров Игорь Иванович, кандидат физико-математических наук, доцент, заведующий лабораторией валидации программного обеспечения, Университета ИТМО.

Методические указания по выполнению практических работ предназначены для студентов по направлению подготовки 10.04.01 «Информационная безопасность» в качестве учебного пособия для выполнения практических работ по курсу «Моделирование криптосистем» В данном учебном пособии объединены практические задания для изучения современных криптографических алгоритмов на уровне стандартов и дополнительных механизмов для защиты. Практические работы содержат в себе как задания на высокоуровневый анализ стандартов, который сводится к изучению ограничений на параметры криптосистем, так и отдельные примеры криптоанализа для более тщательного тестирования современных стандартов шифрования.

Университет ИТМО – национальный исследовательский университет, ведущий вуз России в области информационных, фотонных и биохимических технологий. Альма-матер победителей международных соревнований по программированию – ICPC (единственный в мире семикратный чемпион), Google Code Jam, Facebook Hacker Cup, Яндекс.Алгоритм, Russian Code Cup, Topcoder Open и др. Приоритетные направления: IT, фотоника, робототехника, квантовые коммуникации, трансляционная медицина, Life Sciences, Art&Science, Science Communication. Входит в ТОП-100 по направлению «Автоматизация и управление» Шанхайского предметного рейтинга (ARWU) и занимает 74 место в мире в британском предметном рейтинге QS по компьютерным наукам (Computer Science and Information Systems). С 2013 по 2020 гг. – лидер Проекта 5–100.

© Университет ИТМО, 2022
© Таранов С.В., 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Криптографические методы защиты информации являются критической и ключевой структурой в современных технологиях обеспечения информационной безопасности. Криптография основана на доверии только к сертифицированным реализациям криптосистем, и строго не рекомендуется заниматься собственной разработкой криптографических систем защит без глубоко понимания принципов шифрования информации. Такой подход к разработке криптосистем иногда приводит к негативным результатам, например, разработчики современных программно-аппаратных систем не считают, что для использования и включения криптобиблиотек в свою систему нужна тщательная повторная верификация всего комплекса. Ошибки при интеграции криптопримитивов, их неправильная настройка и использование могут привести к проблемам информационной безопасности не меньше, чем ошибки в самих криптобиблиотеках.

Методические указания по выполнению практических работ предназначены для студентов по направлению подготовки 10.04.01 «Информационная безопасность» в качестве учебного пособия для выполнения практических работ по курсу «Моделирование криптосистем» В данном учебном пособии объединены практические задания для изучения современных криптографических алгоритмов на уровне стандартов и дополнительных механизмов для защиты. Практические работы содержат в себе как задания на высокоуровневый анализ стандартов, который сводится к изучению ограничений на параметры криптосистем, так и отдельные примеры криптоанализа для более тщательного тестирования современных стандартов шифрования.

1. Анализ исторических шифров с помощью программного средства Cryptool 2

Цель практической работы: с помощью программного средства Cryptool 2 изучить принципы работы исторических шифров, а также провести их криптоанализ.

Задачи практической работы

Используя функции программы Cryptool 2, проанализировать следующие криптографические примитивы:

1. Шифр Цезаря, шифры перестановки и замены (как примеры моноалфавитных шифров);
2. Шифр Виженера (как пример полиалфавитного шифра);
3. Структуру и процесс шифрования в роторной машине Энигма.

Изучение каждого криптографического алгоритма предлагается проводить по следующему плану исследования:

1. Выполнить шифрование своего примера открытого текста с собственными настройками криптосистемы. Получить шифротекст, соответствующий заданному входному значению и ключу. Изучить настройки криптосистемы, а именно: возможное пространство ключей, дополнительные механизмы для шифрования и повышения криптостойкости, ограничения и требования к параметрам криптоалгоритма (ограничения на длину криптографического ключа, алфавит открытого текста и шифротекста, слабые параметры, обязательные и рекомендуемые требования к настройке криптосистемы).
2. Выполнить дешифрование закрытого текста с помощью криптографического ключа из первого пункта плана. Оценить, насколько сложно выполнить атаку на ключ методом перебора. Проанализировать сложность процесса шифрования и настройки криптосистемы для использования (шифрования/дешифрования).
3. Выполнить простейший криптоанализ, используя шаблоны из программы Cryptool 2. При криптоанализе можно использовать шифротекст или несколько шифротекстов, полученных в первом пункте практического задания. При криптоанализе считается, что криптографический ключ неизвестен, более того, что нет никакой коррелирующей информации о ключе. Шифротексты без криптографических ключей можно получить на этапе шифрования криптосистемы в Cryptool 2, скрыв предварительно поле для ввода ключа.

Порядок выполнения работы

1. Выполнить шифрование открытого текста, основываясь на примере криптосистемы, реализованной в программном средстве CrypTool 2. Для того чтобы открыть соответствующий шаблон, в программном средстве перейдите в Раздел Templates в подкаталог Cryptography -> Classical -> Caesar Cipher (Рисунок 1.1).

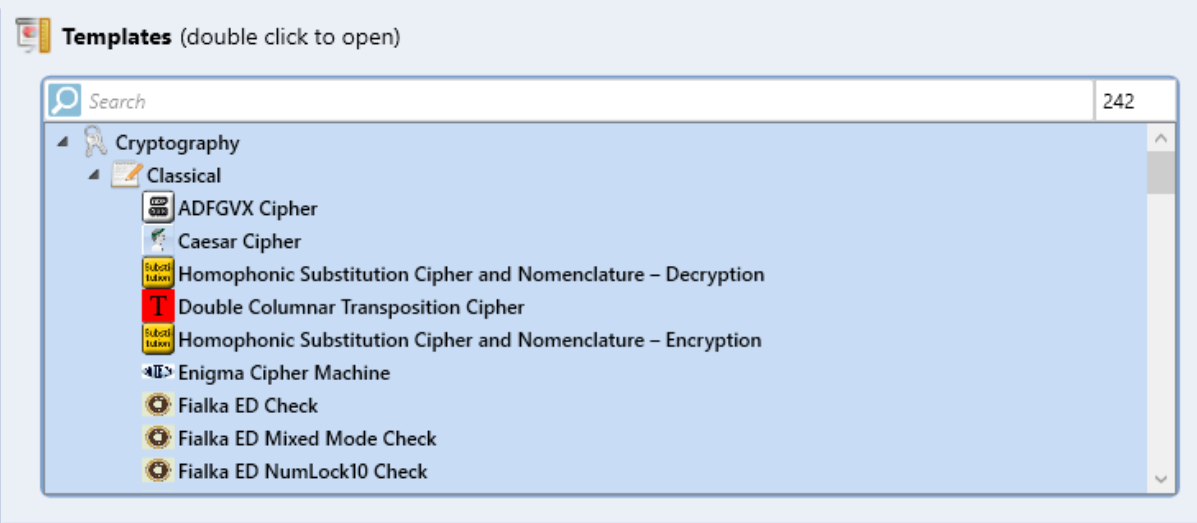

Рисунок 1.1. Шаблоны классических криптосистем в программном средстве Cryptool 2

2. Открыть шаблон для шифрования с помощью шифра Цезаря. Задать собственный открытый текст в поле Plaintext, разрешенный алфавит в поле Alphabet и значение криптографического ключа (Рисунок 1.2). Открытый текст и ключ выбрать самостоятельно. Для более быстрого и успешного последующего криптоанализа необходимо, чтобы открытый текст был как можно больше, не содержал недопустимых шифром значений (знаков пунктуации, скобок, вставок из другого алфавита и пр.). Сохранить получившийся шифротекст из поля Ciphertext и значение ключа для дальнейшего использования.

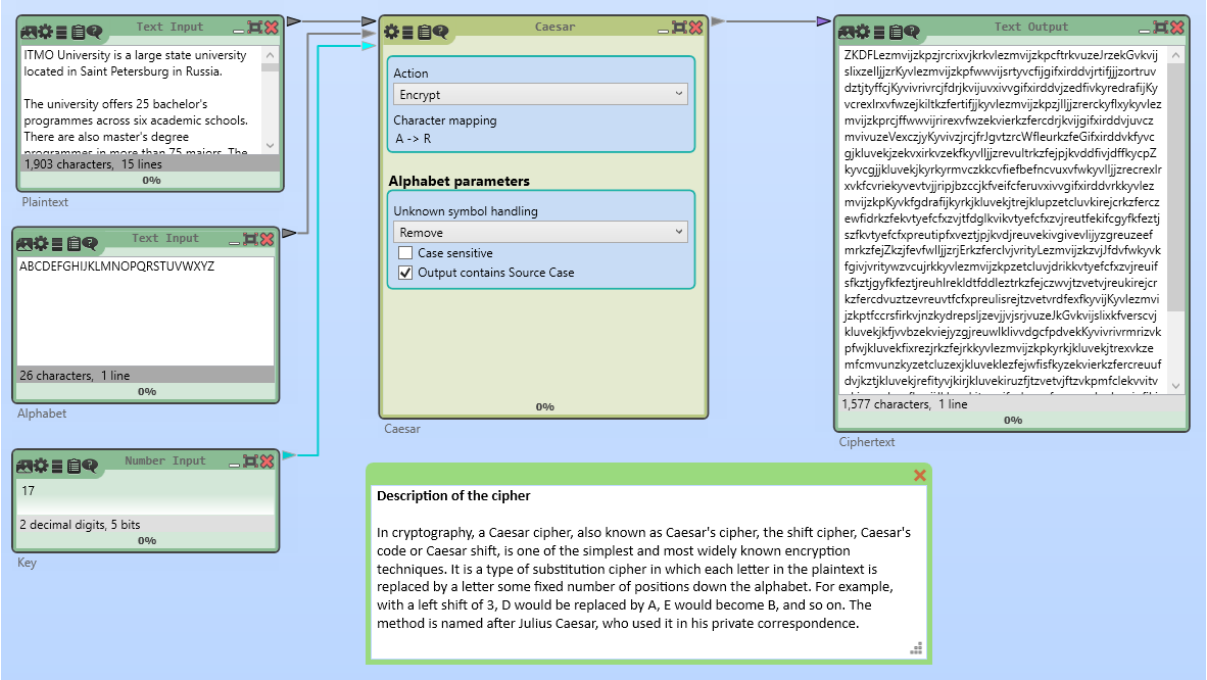

Рисунок 1.2. Схема шифрования с помощью моноалфавитного шифра Цезаря.

3. Выполнить дешифрование с помощью того же шаблона, выбрав в блоке Caesar режим дешифрования (Рисунок 1.3). Проанализировать выходное значение. Совпадает ли оно полностью с открытым текстом из пункта 2?

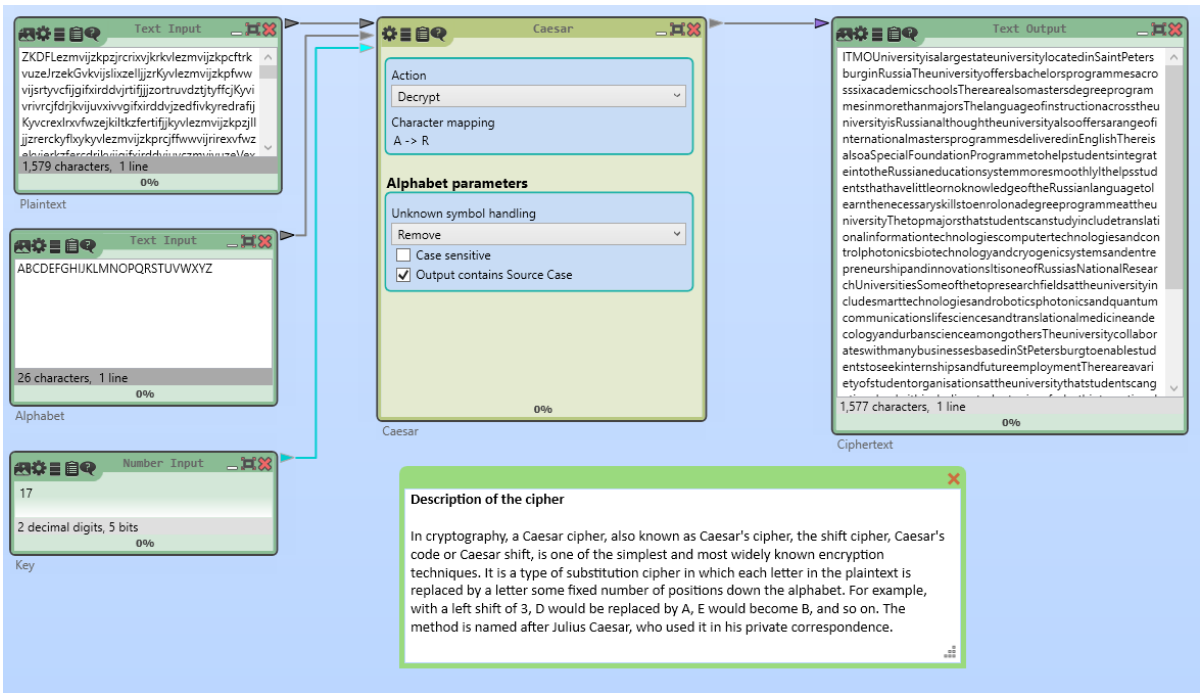

Рисунок 1.3. Схема дешифрования шифра Цезаря.

4. Выполнить простейший криптоанализ, используя шаблоны из программы Cryptool 2. Шаблон для атаки на основе частотного анализа находится по пути Templates -> Cryptoanalysis -> Classical -> Caesar Analysis using character frequencies (Рисунок 1.4). Целью криптоанализа является восстановление секретного ключа на основе закрытого текста. В качестве входных параметров предлагается указать закрытый текст, который был получен в пункте 2, и попытаться восстановить открытый текст и секретный ключ. Проанализировать путем генерации и шифрования других шифротекстов то, как влияют на успешность криптоанализа следующие параметры:
    - Длина открытого текста;
    - Удаление специальных символов и вставок, которые не входят в алфавит шифра (формул, кавычек, обозначений переменных и пр.);
    - Удаление пробелов между словами.

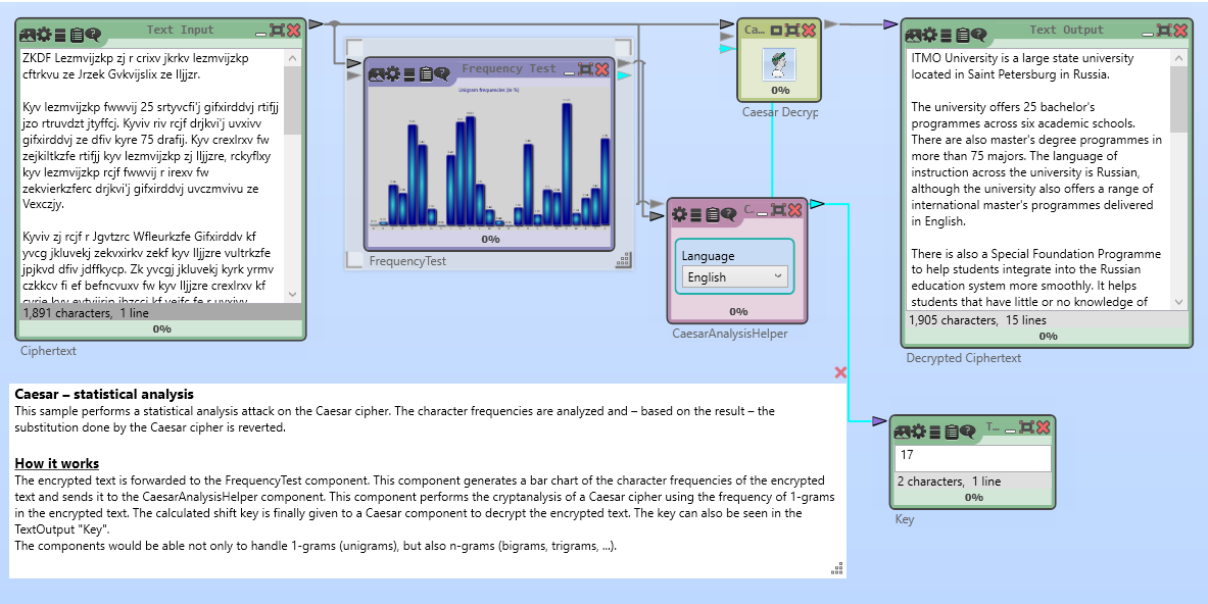

Рисунок 1.4. Частотный анализ моноалфавитных шифров.

5. Выполнить криптоанализ шифра Цезарь, используя шаблон, реализующий атаку полным перебором. Путь до шаблона Templates -> Cryptoanalysis -> Classical -> Caesar Brute Force Analysis.
6. По аналогии с пунктами 1-5 провести анализ других моноалфавитных и полиалфавитных шифров из практической работы, а также провести атаки на каждый криптоалгоритм. Необходимые шаблоны Cryptool 2 (Templates -> Cryptoanalysis/Cryptography -> Classical):
   для шифров:
    - Substitution Cipher;
    - Transposition Cipher;

- Vigenère Cipher.

для атак на шифры:

- Transposition Brute-Force Analysis;
- Transposition Crib Analysis;
- Transposition Genetic Analysis;
- Transposition Hill Climbing Analysis;
- Monoalphabetic Substitution Analyzer;
- Frequency Analysis;
- Vigenère Analysis.

7. Выполнить пункты 1-5 для эмулятора роторной машины Энигмы. Необходимые шаблоны Cryptool 2 (Templates -> Cryptoanalysis/Cryptography -> Classical):
   для Энигмы:
   - Enigma Cipher Machine.

   для атак на роторную машину Энигма:
   - Enigma Gillogly Attack;
   - Enigma Hillclimbing Attack;
   - Enigma Simulated Annealing Attack;
   - Enigma Turing Bombe Attack.

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;
- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

2. Основные структурные элементы блочного симметричного алгоритма DES

Цель: изучить основные принципы работы алгоритмы DES.

Задачи практической работы

1. Проанализировать эмуляцию алгоритма DES и примитивных атак на шифр, используя Cryptool 2. Выделить основные необходимые настройки шифра и требуемые ограничения на параметры.
2. Выполнить 1 цикл раундовой функции алгоритма DES вручную, то есть выполнить все функции, входящие в раундовую функцию DES для фиксированного входного двоичного вектора с отображением промежуточных значений шифрования. Также для подробного изучения шифра может быть использована программная реализация 1 раунда (или полной системы) DES в режиме отладки с выводом промежуточных значений шифрования.
3. Проанализировать принципы использования криптосистем в современных приложениях на примере библиотеки OpenSSL.

Порядок выполнения работы

1. Для визуализации алгоритма DES предлагается использовать шаблон Cryptool 2 Templates -> cryptography -> modern -> symmetric-> DES Visualization (Рисунок 2.1). Укажите в данном шаблоне свои входные данные и криптографический ключ

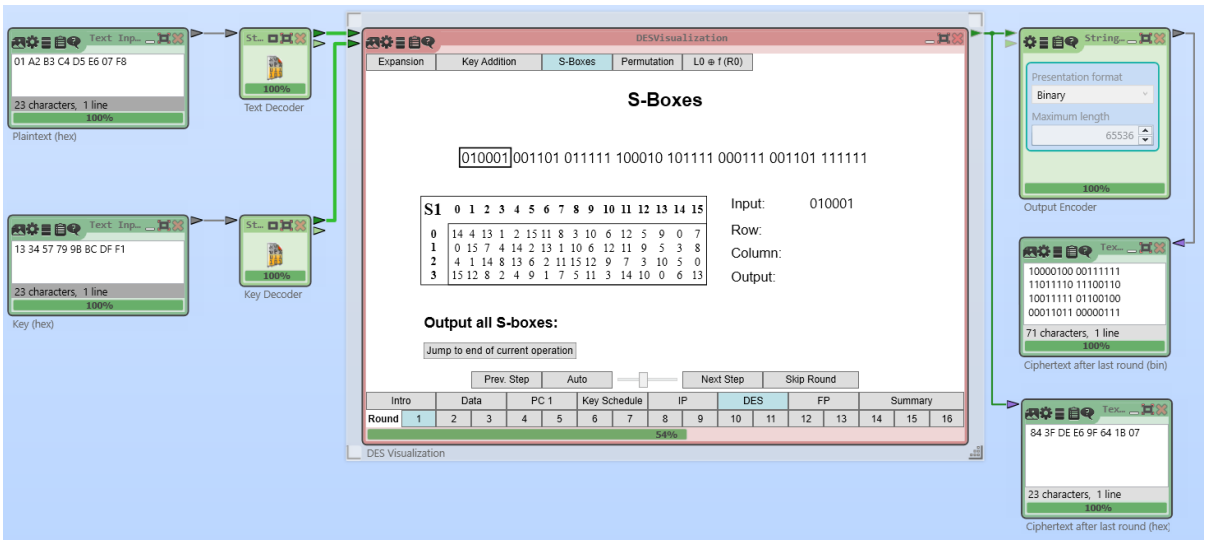

Рисунок 2.1. Визуализация блочного симметричного шифра DES

Проследите за поэтапным выполнением процедуры шифрования криптоалгоритма DES. Особое внимание уделите функции расширения ключа, структуре сети Фейстеля для алгоритма DES, этапам раундовой функции (перестановка с расширением E, сложение с ключом, нелинейная замена, прямая перестановка P). Отметьте, как изменяется длина подблоков после каждой итерации.

2. Запустить визуализацию «лавинного эффекта» DES templates -> cryptoanalysis -> modern -> avalanche (DES) (Рисунок 2.2). Проследить за лавинным эффектом в процессе шифрования DES для своих значений открытого текста и ключа.

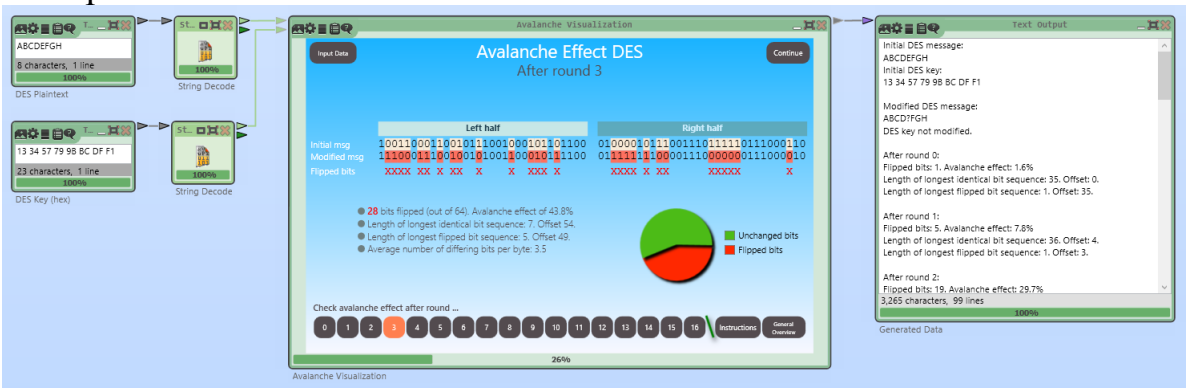

Рисунок 2.2. Лавинный эффект блочного симметричного шифра DES

Отметить процент искаженных битов в результате шифрования, скорость лавинного эффекта, характер группировки искаженных значений. Внести искажения в несколько битов и проанализировать характер лавинного эффекта. Внести искажения в промежуточные значения шифрования на 10, 12, 14 раундах, проследить за тем, каких значений достигнет лавинный эффект.

3. Изучить принципы работы различных режимов шифрования блочных симметричных шифров, используя шаблон Cryptool 2 Templates -> cryptography -> modern ->Block Modes of Symmetric Ciphers (Рисунок 2.3).

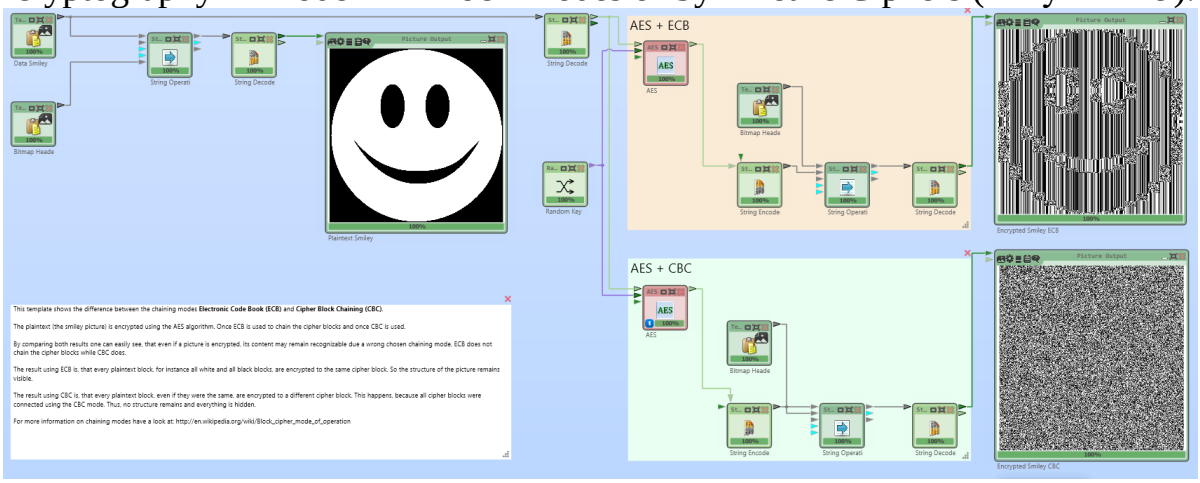

Рисунок 2.3. Режимы шифрования блочных симметричных шифров.

Отметить возможные уязвимости при использовании режима шифрования ECB с точки зрения информационной безопасности. Изучить другие режимы шифрования и способы сцепления блоков (режимы CBC, CFB, OFB, RD, CTR, GCM).

4. Провести padding oracle атаку на алгоритм DES для своих значений (Рисунок 2.4). Выделить этапы проведения атаки. Привести примеры применения данной атаки в реальной жизни.

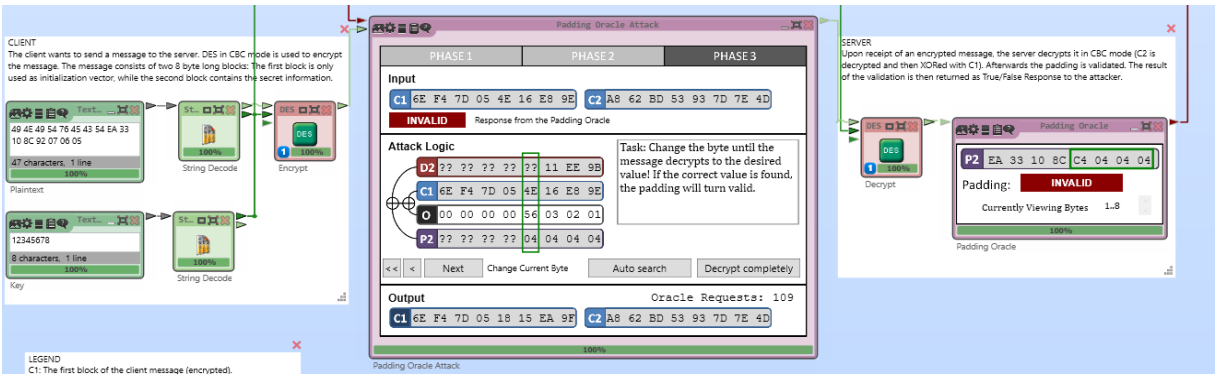

Рисунок 2.4. Padding oracle атака на шифр DES

5. Выполнить 1 цикл раундовой функции алгоритма DES вручную. Сгенерировать случайное 32-битное значение входного сообщения и 48-битный раундовый ключ. Продемонстрировать как выполняются следующие операции алгоритма DES:
   - перестановка с расширением E (Рисунок 2.5);

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Рисунок 2.5. Перестановка с расширением алгоритма DES

   - побитовое сложение с ключом;
   - операция подстановки S-box (Рисунок 2.6);

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	5	13

Рисунок 2.6. Пример операции подстановки из алгоритма DES

- прямая перестановка P (Рисунок 2.7).

P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Рисунок 2.7. Прямая перестановка раундовой функции алгоритма DES

Опишите, как использовать представленные выше таблицы замены и перестановки. Результат выполнения операций для своих входных значений можно представить в виде:

- листов с отсканированным отчетом, на котором представлено рукописное выполнение необходимых этапов шифрования;
- листов отчета с этапами шифрования в электронном виде, который выполнен в любом удобном редакторе текстовых документов;
- листов с программным кодом или псевдокодом, которые выполняют вышепредставленные этапы алгоритма.

6. Выполнить шифрование файла с помощью криптографической библиотеки OpenSSL. Для этого с помощью команды *openssl enc -list* выведите список доступных криптографических стандартов для шифрования

```
OpenSSL> enc -list
Supported ciphers:
-aes-128-cbc               -aes-128-cfb               -aes-128-cfb1
-aes-128-cfb8              -aes-128-ctr               -aes-128-ecb
-aes-128-ofb               -aes-192-cbc               -aes-192-cfb
-aes-192-cfb1              -aes-192-cfb8              -aes-192-ctr
-aes-192-ecb               -aes-192-ofb               -aes-256-cbc
-aes-256-cfb               -aes-256-cfb1              -aes-256-cfb8
-aes-256-ctr               -aes-256-ecb               -aes-256-ofb
-aes128                    -aes128-wrap               -aes192
-aes192-wrap               -aes256                    -aes256-wrap
-aria-128-cbc              -aria-128-cfb              -aria-128-cfb1
-aria-128-cfb8             -aria-128-ctr              -aria-128-ecb
```

Рисунок 2.8. Доступные алгоритмы шифрования данных библиотеки OpenSSL

Выбрать по крайней мере три подхода шифрования, включающих алгоритм DES, например, можно выбрать DES в нескольких режимах шифрования или несколько модификаций 3DES.

Задать свое входное значение либо в виде файла, либо в виде сообщения в командной строке и зашифровать их, используя 3 выбранных подхода шифрования с DES (Рисунки 2.9-2.11)

```
OpenSSL> enc -des -e -in example.txt -out example.txt.enc
enter des-cbc encryption password:
Verifying - enter des-cbc encryption password:
```

Рисунок 2.9. Пример шифрования DES из библиотеки OpenSSL

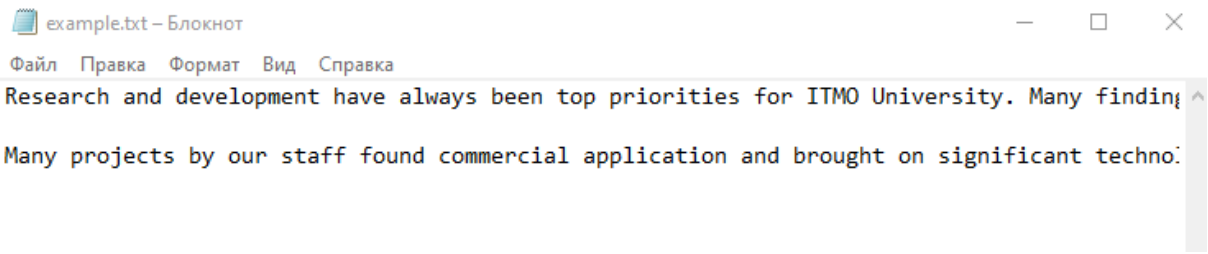

Рисунок 2.10. Пример открытого текста для шифрования с помощью библиотеки OpenSSL

Файл example.txt.enc, полученный после шифрования

```
Salted__Ц эм¬9,▯%ҐСЯІ•—U1  СЯ^ҐХЃasц /ЩqMSЮРѓA d▯▯6▯ЗЛ]
SЖ7Ґ;ДDu▯Cd"ъWj·CI▯Нп▯ЧЯн¦у▯'▯4Ж!ђнт▯▯\MI+iЯВ▯EL▯иб▯
;d—ъ‡ IfS▯—xuA_ѓ▯E4sP‹ђi1™▯▯—дЖў8bэПЦu ng▯¤§utkoќ ЩZщ▯JE
¤▯▯u@:Ц‹ A{+МЮ-#jfЖIФЫ▯Ъ0‹Ё oЋ§ЯЁй@iБ!OLђЩп ▯ L▯п(Ц4 З†%-
Пj6Oґ,AIk‘RAЃH&Vз‘°Ьж|Ы▯ЪУчцК▯µAx ▯zH9¶jЇ0Ш= J‹пL;E jLЇa&ў
µё ¤©ЕН-ђLЁ▯ PaO;рсбшаТ7  OIБ0„fCшщ▯ЦтцяЫю¶R†}XV%Ъ iзЗS·
Ёjѓпu  ZЕиTi!ЁJe|њЋ»ДF.Ц▯WдхВ#U▯s▯▯Г7_Инь&”fЪ¬P¤хX
‘+ф[HmI,свС▯8КаЫ№"$съЮМ„©оЫЎ▯Бљ°’j! оч ж,™uапY**‹уБЄж5µц Мь—
Р IН▯▯цф ▯ ?▯ЁЬdт<=-!»▯+вФdп%LВЊвй▯ЮU=WиУr",▯sНЇ¶~љЦ¦▯™▯9°н
```

Рисунок 2.11. Шифротекст, полученный в результате обработки библиотекой OpenSSL

7. Выполнить дешифрование файла из предыдущего пункта. Проверить получившийся открытый текст на ошибки. Проанализировать следующие флаги и дополнительные параметры шифрования DES в OpenSSL: -salt, -a, -k, -iter. Как эти опции влияют на криптостойкость шифра?

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;

- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

3. Основные структурные элементы алгоритма AES

Цель: изучить основные принципы работы алгоритма AES.

Задачи практической работы

1. Проанализировать эмуляцию алгоритма AES и примитивных атак на шифр, используя Cryptool 2. Выделить основные необходимые настройки шифра и требуемые ограничения на параметры.
2. Выполнить 1 цикл раундовой функции алгоритма AES вручную. Отразить промежуточные результаты шифрования после всех этапов алгоритма AES. Дать математическое обоснование для каждой операции. Также для подробного изучения шифра может быть использована программная реализация 1 раунда (или полной системы) AES в режиме отладки с выводом промежуточных значений шифрования.
3. Проанализировать принципы использования криптосистемы в современных приложениях на примере библиотеки OpenSSL.

Порядок выполнения работы

1. Для визуализации алгоритма AES предлагается использовать шаблон Cryptool 2 Templates -> cryptography -> modern -> symmetric-> AES Visualization (Рисунок 3.1). Укажите в данном шаблоне свои входные данные и криптографический ключ.

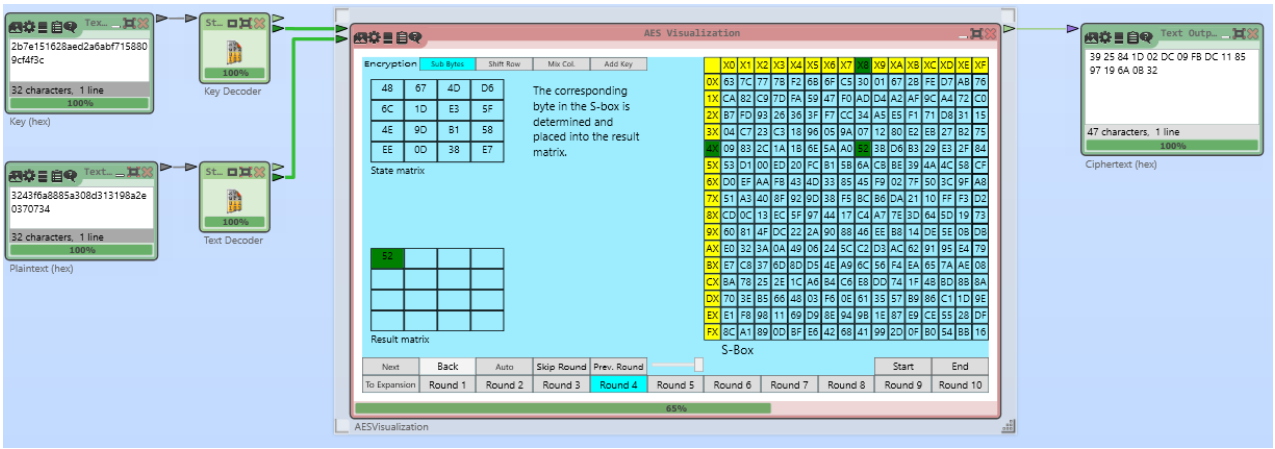

Рисунок 3.1. Визуализация алгоритма блочного симметричного шифра AES

Проследите за каждым этапом выполнения процедуры шифрования криптоалгоритма AES. Рассмотрите подробно следующие операции:

- KeyExpansion;
- AddRoundKey;

- ShiftRows;
- MixColumn;
- SubBytes.

Обратите также внимание на то, как заполняется матрица состояний до начала работы шифра. Изучите последовательность выполнения операций при шифровании и дешифровании. Все ли раунды AES состоят из одинаковых операций? Проанализируйте то, как каждое значение матрицы состояний меняется по результатам раунда AES.

2. Запустить визуализацию «лавинного эффекта» AES templates -> cryptoanalysis -> modern -> avalanche AES (Рисунок 3.2). Проследить за лавинным эффектом в процессе шифрования AES для своих значений открытого текста и ключа.

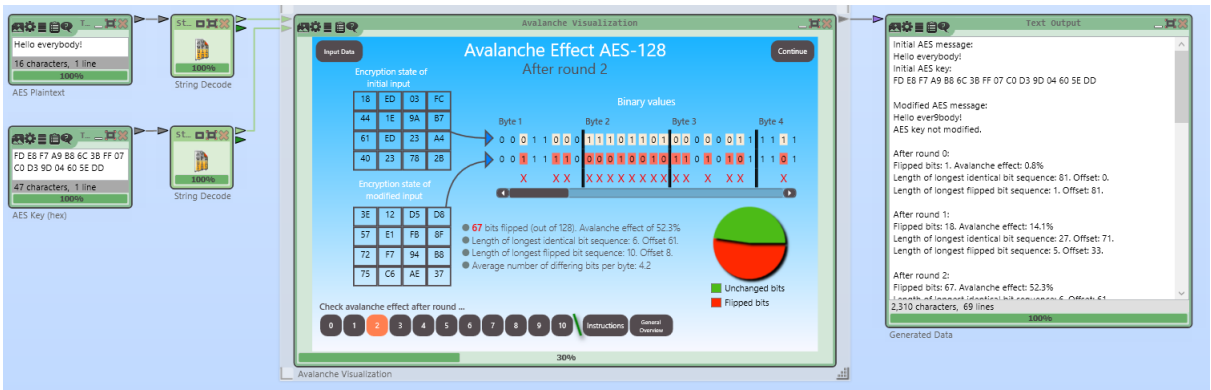

Рисунок 3.2. Визуализация лавинного эффекта алгоритма AES

Отметьте процент искаженных битов в результате шифрования, скорость лавинного эффекта, как сгруппированы искаженные значения. Внесите искажения в несколько битов и проанализируйте характер лавинного эффекта.

3. Провести атаку на основе известного открытого текста (шаблон templates->cryptanalysis->modern->AES known-plaintext analysis) на алгоритм AES для своих значений (Рисунок 3.3). Выделить этапы проведения атаки. Описать ограничения, при которых рассматриваемая атака на криптоалгоритм становится возможной. Привести примеры проведения данной атаки на реальных системах передачи, хранения и обработки информации.
4. Изучить основные этапы проведения дифференциального анализа на блочные симметричные криптосистемы на примере шаблонов Differential Cryptanalysis Tutorial (Рисунок 3.4).

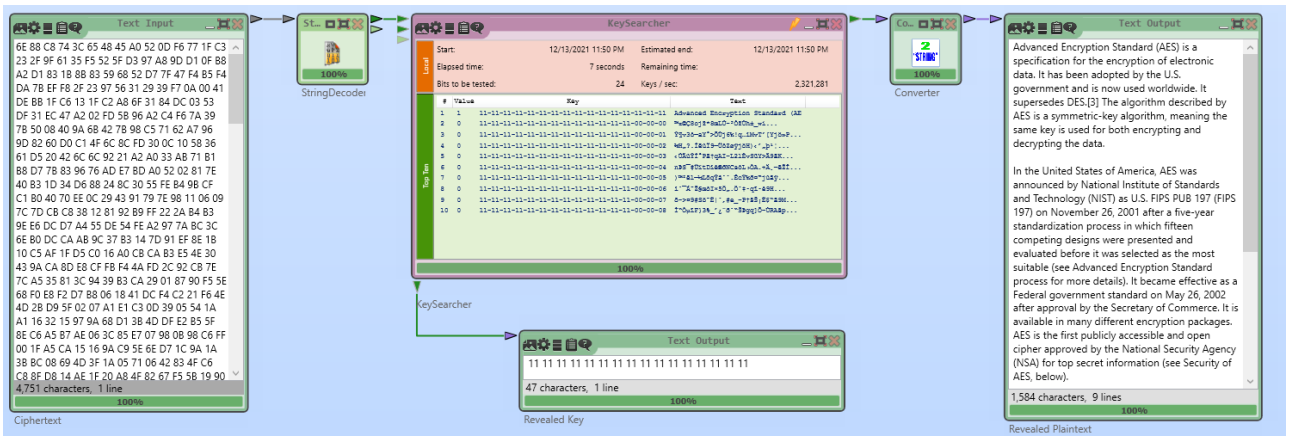

Рисунок 3.3. Атака на основе известного текста на алгоритм AES.

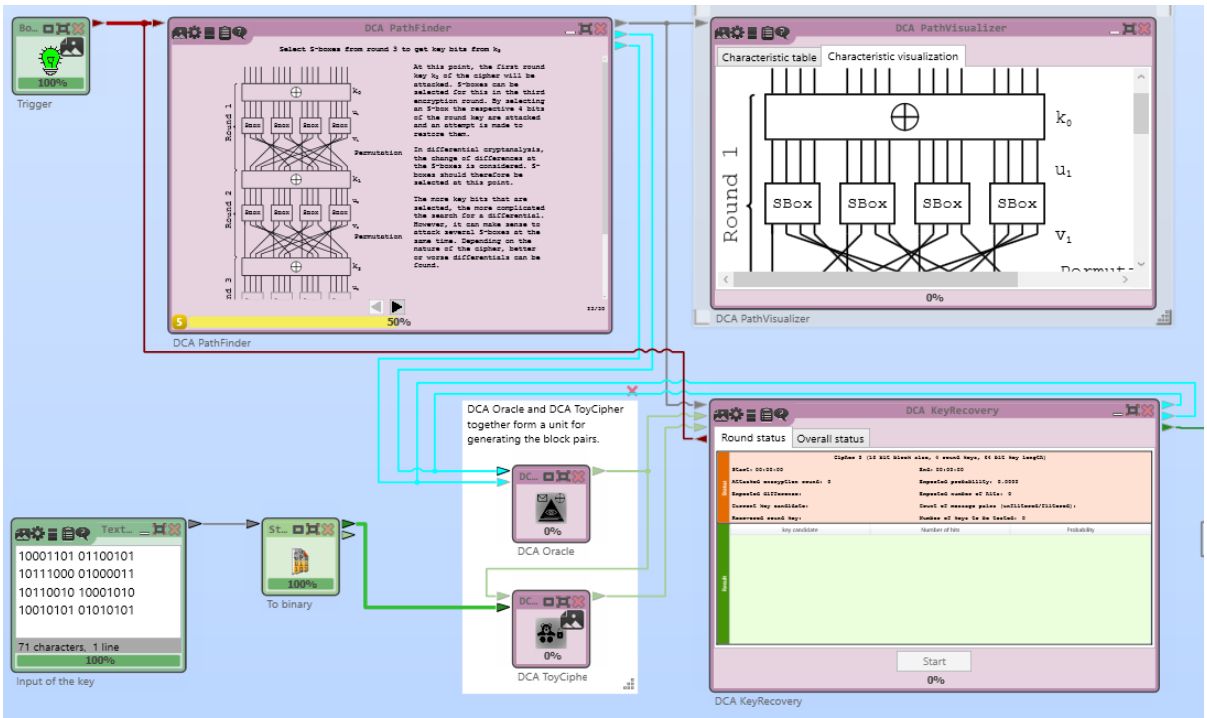

Рисунок 3.4. Дифференциальный анализ алгоритма AES

Описать в отчете, как происходит анализ таблиц нелинейной замены. Дать краткое пояснение, что такое дифференциал и как он используется для подбора криптографического ключа.

5. Выполнить 1 цикл раундовой функции алгоритма AES вручную. Для этого необходимо сгенерировать случайное 128-битное значение входного сообщения и 128-битный раундовый ключ. Продемонстрировать как выполняются следующие операции алгоритма AES:
   - Процедура расширения криптографического ключа. Описать сколько 32 битных слов генерируется для каждой длины криптографического ключа (128, 192, 256). Показать как слова, составляющие ключ,

получаются из частей мастер ключа (операции, преобразования и пр.).

- Операция ShiftRows (Рисунок 3.5).

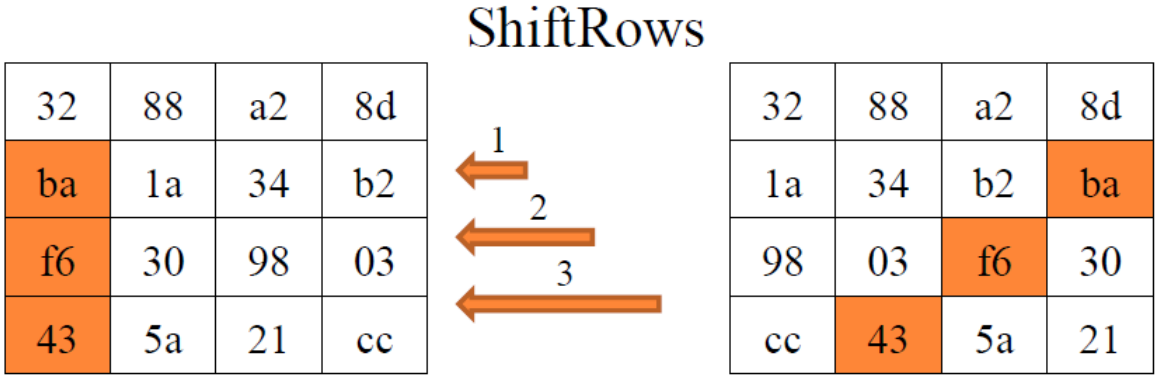

Рисунок 3.5. Операция ShiftRows

- Побитовое сложение с ключом.
- Операция MixColumn (Рисунок 3.6).

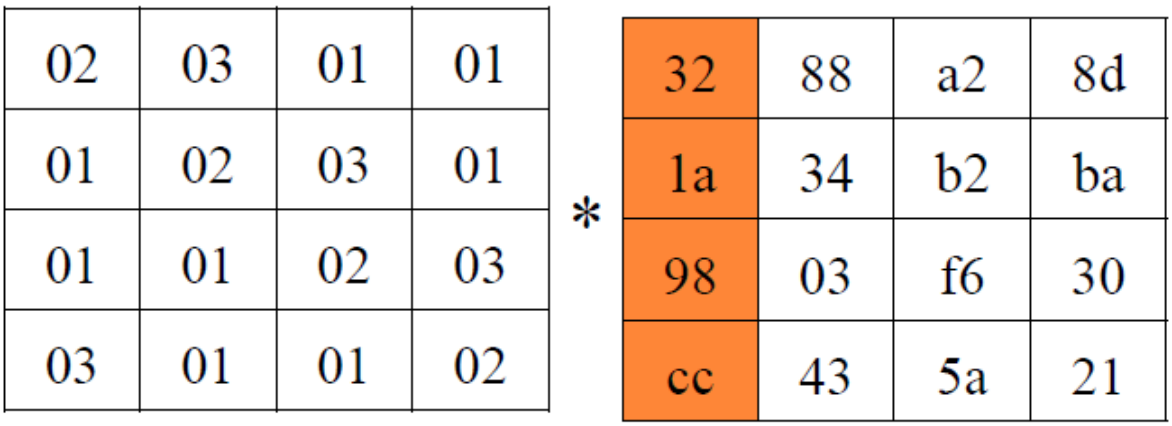

Рисунок 3.6. Операция перемешивания столбцов MixColumn

При рассмотрении операции MixColumn описать, как происходит преобразование шестнадцатеричных значений в элемент расширенного поля Галуа GF(2^8). В отчете дать подробное описание процесса умножения элементов расширенного поля Галуа. Рассмотреть отдельно случай, когда при произведении элементов возможно переполнение, например, при произведении 02*FF. Дать описание операции InversionMixColumn, которое выполняет обратное действие для MixColumn. Какая матрица коэффициентов будет использоваться в качестве первого операнда при умножении матриц в InversionMixColumn? Можно ли операцию InversionMixColumn заменить несколькими операциями MixColumn?

- Операция SubBytes (Рисунок 3.7)

	y															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
x 8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Рисунок 3.7. Операция замены байтов алгоритма AES

Описать, как работают представленные выше операции замены и перестановки. Результат выполнения операций для своих входных значений можно представить в виде:

- листов с отсканированным отчетом, на котором представлено рукописное выполнение необходимых этапов шифрования;
- листов отчета с этапами шифрования в электронном виде, который выполнен в любом удобном редакторе текстовых документов;
- листов с программным кодом или псевдокодом, которые выполняют вышепредставленные этапы алгоритма.

6. Выполнить шифрование файла с помощью криптографической библиотеки OpenSSL. Для этого с помощью команды *openssl enc -list* выведите список доступных криптографических стандартов для шифрования (Рисунок 3.8).

Выберите по крайней мере три подхода шифрования, включающих алгоритм AES, например, можно выбрать AES в нескольких режимах шифрования или с разными значениями криптографического ключа.

```
OpenSSL> enc -list
Supported ciphers:
-aes-128-cbc               -aes-128-cfb               -aes-128-cfb1
-aes-128-cfb8              -aes-128-ctr               -aes-128-ecb
-aes-128-ofb               -aes-192-cbc               -aes-192-cfb
-aes-192-cfb1              -aes-192-cfb8              -aes-192-ctr
-aes-192-ecb               -aes-192-ofb               -aes-256-cbc
-aes-256-cfb               -aes-256-cfb1              -aes-256-cfb8
-aes-256-ctr               -aes-256-ecb               -aes-256-ofb
-aes128                    -aes128-wrap               -aes192
-aes192-wrap               -aes256                    -aes256-wrap
-aria-128-cbc              -aria-128-cfb              -aria-128-cfb1
-aria-128-cfb8             -aria-128-ctr              -aria-128-ecb
```

Рисунок 3.8. Доступные алгоритмы шифрования данных.

Задать свое входное значение либо в виде файла, либо в виде сообщения в командной строке и зашифровать их, используя 3 выбранных подхода шифрования с AES (Рисунок 3.9-3.11).

```
OpenSSL> enc -aes-128-cfb -e -in example.txt -out example2.txt.enc
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
```

Рисунок 3.9. Шифрование алгоритмом AES с помощью библиотеки OpenSSL

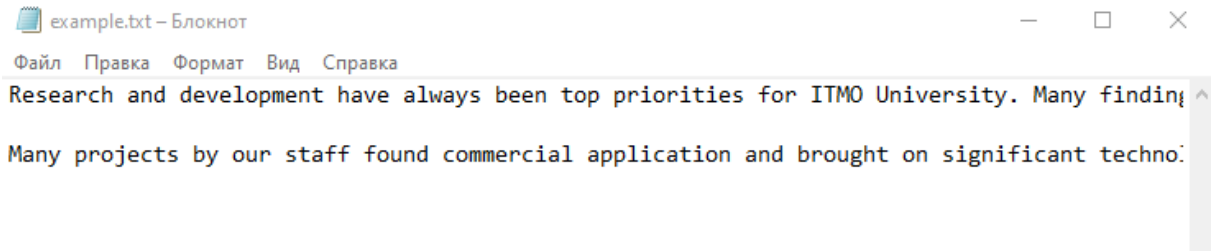

Рисунок 3.10. Пример исходного текста для шифрования алгоритмом AES.

```
Salted__ †чiZЦ  я-ЖщК9кч_‡µBW уэgрЖ•/#ЊµЄSsCш їGЄP7▯ «+±V▯Ь‡>
X"‡Љ©юШsI¬▯ єш€▯ O"§-цL/J°Q° |A}kЊ▯a9     †,ва%#4к«м9d(}▯ "2Ќ▯ eэiЁайс
K!s▯  ё▯€‹yI·WI▯ ^▯¬▯ 'гиЭ2·њ1цµ▯ РЎ▯ К_  ЊН:3НХU°щfмr§cУы©# F r▯ jњЕ„±
Ц M([”µ»KJkBЁїм‹€БТъ’Y%Ў"Jтчњ▯ Z`1ёh|N”Ъє%†
▯ГА»▯ЛТ ▯eh•НУ-▯ о€±ЖЯ▯ хљэЧїжСµТ▯▯ њk§▯ \Ф|ЇЯмњєн▯з@YЎ▯ -▯ХсВґ▯Иb·±--
(ОЛНМэ”СУМљЬ▯ Dn#i(ЧЬц*¦ё=±"uРШйЮЗWЙкэ+1ы•¬ П▯ юЎБЮў]eэD”-
X~,Ћж’fSњ iЃ‰Lg▯ L▯ ^ы▯▯%izN▯Ѓ▯ ЁrH=  `▯e▯ Ќ?;fPt]„x};ЊAЂъ©ље;
°ї)б▯  3РТ·ќЯ,.чЭ8iб
БМЄ ўЃ§ЭТ~Ґ:,ЪфйU[N3-грБ6Лѓ J▯ &Ґ $±Ў€{D{ДШj]d_|и}▯ Т▯ Ё▯ 0ЦШ"ЧЎ▯ ТЙiс}
Е¦ев`Ц7Р1Vж3w3IXhк\°6Х¶  лЙ▯▯ ·[Ч2Оj5Ўi+UЪьШ▯ :п▯▯‹НбСN?-
рч Єн:КЇ+z#cРЪ‹МТѓ УА ,§°иђВ@▯ #▯©v§t¤0їMPzR=ўпў-тjL▯жќ-›Шк>·
¶Ya,▯ C&aCФ-e’aC▯ ?д▯ eSXэтj©ж4чб юz.▯mA»5j9C@▯ *Ъдb’ьу
```

Рисунок 3.11. Файл example2.txt.enc, полученный после шифрования

7. Выполнить дешифрование файла из предыдущего пункта. Проверить получившийся открытый текст на ошибки. Проанализировать следующие флаги и дополнительные параметры шифрования AES в OpenSSL: -salt, -a, -k, -iter. Как эти опции влияют на криптостойкость шифра?

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;
- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

4. Асимметричные криптосистемы

Цель: изучить основные принципы работы асимметричных криптосистем на примере алгоритма RSA.

Задачи практической работы

1. Проанализировать эмуляцию алгоритма RSA и примитивных атак на шифр, используя Cryptool 2. Выделить основные необходимые настройки шифра и требуемые ограничения на параметры.
2. Программно реализовать и модифицировать любую асимметричную криптосистему. В случае отсутствия опыта программирования подойдет реализация алгоритма на псевдокоде или в виде блок-схем, включающих основные этапы алгоритма с отображением формул и основных математических действий. Атаки и модификации, приведенные ниже, указаны для RSA. Если атака или модификация неприменима для реализуемого алгоритма, разрешается найти любую альтернативу (атаки, применимой к алгоритму; модификации для ускорения алгоритма и дополнительной защиты).
3. Для созданной реализации криптосистемы провести примитивный криптоанализ на устойчивость к описанным в пункте 4 атакам, а также сделать минимальные модификации по оптимизации (ускорению процессов шифрования, дешифрования, процесса генерации ключей).

Порядок выполнения работы

1. Провести визуализацию алгоритма RSA с помощью следующего шаблона: Cryptool 2 Templates -> cryptography -> modern -> asymmetric-> RSA Cipher (Рисунок 4.1). Указать в данном шаблоне свои входные данные и криптографический ключ.

Проанализировать каждый из блоков шаблона. Задать параметры ключевой пары RSA вручную, для этого открыть настройки блока RSA KeyGenerator и указать собственные значения параметров N, e, d (Рисунок 4.2).

Для блоков RSA Encryption и RSA Decryption укажите, как выглядят формулы для шифрования и дешифрования с учетом созданных ранее ключей.

2. Выполнить атаку на алгоритм RSA на основе общего делителя для модуля шифрования (Рисунок 4.3). Выделить основные этапы проведения атаки, при каких условиях атака возможна.

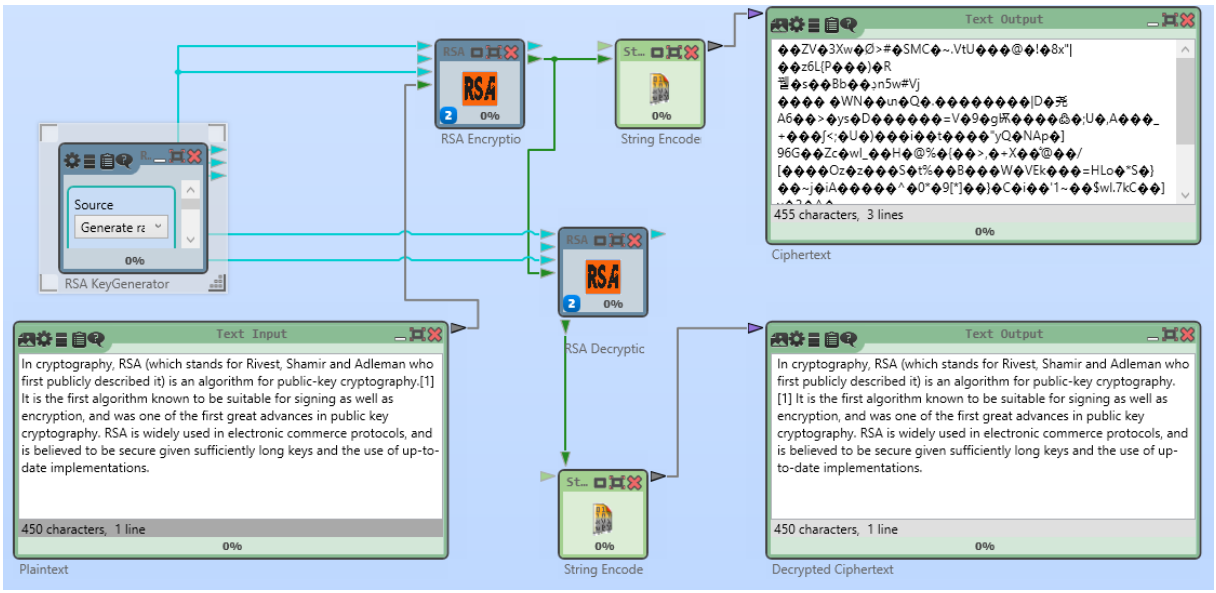

Рисунок 4.1. Процессы шифрования и дешифрования в асимметричной криптосистеме RSA

Рисунок 4.2. Настройка параметров асимметричной криптосистемы RSA.

Зашифровать два шифротекста с помощью различных модулей шифрования N с общим делителем. Показать, как эти шифротексты могут быть дешифрованы при неизвестных дешифрующих экспонентах d.

3. Создать программную реализацию асимметричной криптосистемы. Ниже будут даны атаки и модификации, предлагаемые для реализации криптосистемы RSA, однако могут быть разработаны любые другие асимметричные криптосистемы (например, криптосистемы Эль-Гамаль, Рабина, Пейэ и пр.). Язык программирования выбирается на усмотрение обсучающегося.

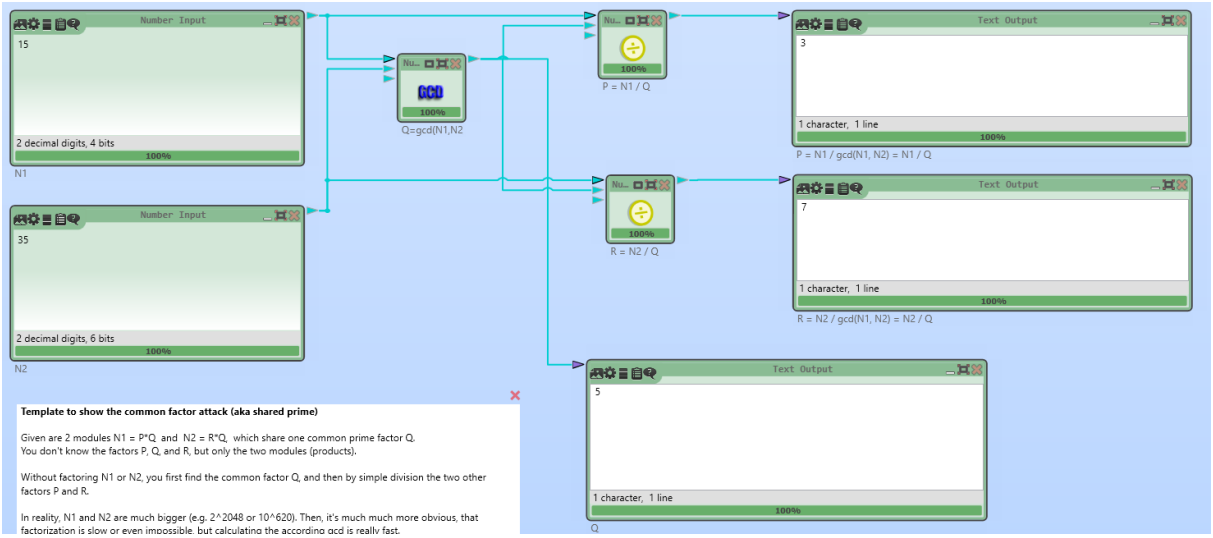

Рисунок 4.3. Атака на основе общего делителя модуля RSA.

4. Модифицировать программную реализацию, созданную в пункте 5, чтобы повысить эффективность процедуры шифрования/дешифрования, генерации ключей, устойчивость к определенным атакам. Предлагается выполнить хотя бы 3 из нижепредставленных модификаций:

   - использование китайской теоремы об остатках для ускорения процесса дешифрования RSA;
   - использование алгоритма быстрого возведения в степень для ускорения процесса шифрования RSA;
   - показать, как используются алгоритмы дополнений (padding) в асимметричных криптосистемах, например, из стандарта PKCS#7;
   - добавить в программную реализацию RSA ограничения на параметры, дополнительные механизмы, тесты и проверки, которые позволят защититься от:
     - атак «человек посередине»;
     - атаки хастада или любой другой атаки на малую экспоненту;
     - атаки по времени выполнения на несбалансированные ветки алгоритма;
     - атаки на RSA при неудачном выборе параметров криптосистемы (модуль N является произведением простого числа Марсенна 8191 и простого числа Ферма 65537, простые числа p и q слишком близки друг к другу и пр.);
     - атаки повторным шифрованием;
     - атаки методом Ферма.

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;
- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

5. Цифровые подписи и сертификаты в GNU Privacy Guard. Система управления ключей Kleopatra

Цель работы: изучение основных функций программного средства шифрования информации, создание цифровых подписей GnuPG, получение навыков работы с системой управления ключей Kleopatra.

Установка GnuPG

Ссылка для скачивания установочных файлов GnuPG для Windows: http://gpg4win.org/download.html.

При установке программного средства GnuPG рекомендуется установить Менеджер ключей Kleopatra, поскольку дальнейшее руководство в данной практической работе будет приведено для связки Kleopatra+GnuPG.

Процедура генерации ключей

Запустить Менеджер ключей Kleopatra (Рисунок 5.1)

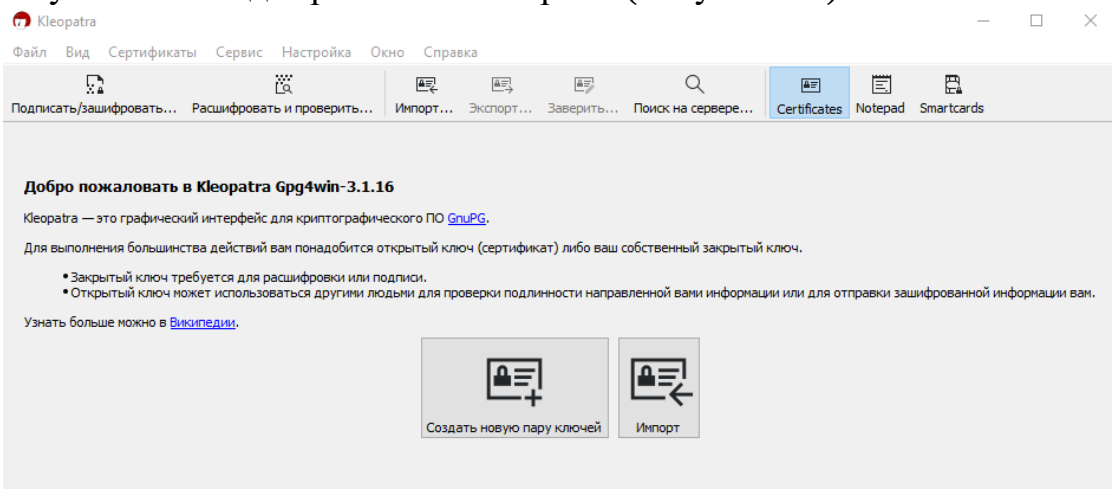

Рисунок 5.1. Менеджер ключей Kleopatra

В главном окне программы, которое представлено на вышеприведённом скриншоте, отображаются известные программе ключи (свои пары ключей или открытые чужие ключи). Открытые ключи в программе включаются в сертификаты.

Чтобы создать новую пару ключей, необходимо выбрать пункт меню ***File → New Certificate***, в результате появится окно, показанное на следующем скриншоте (Рисунок 5.2).

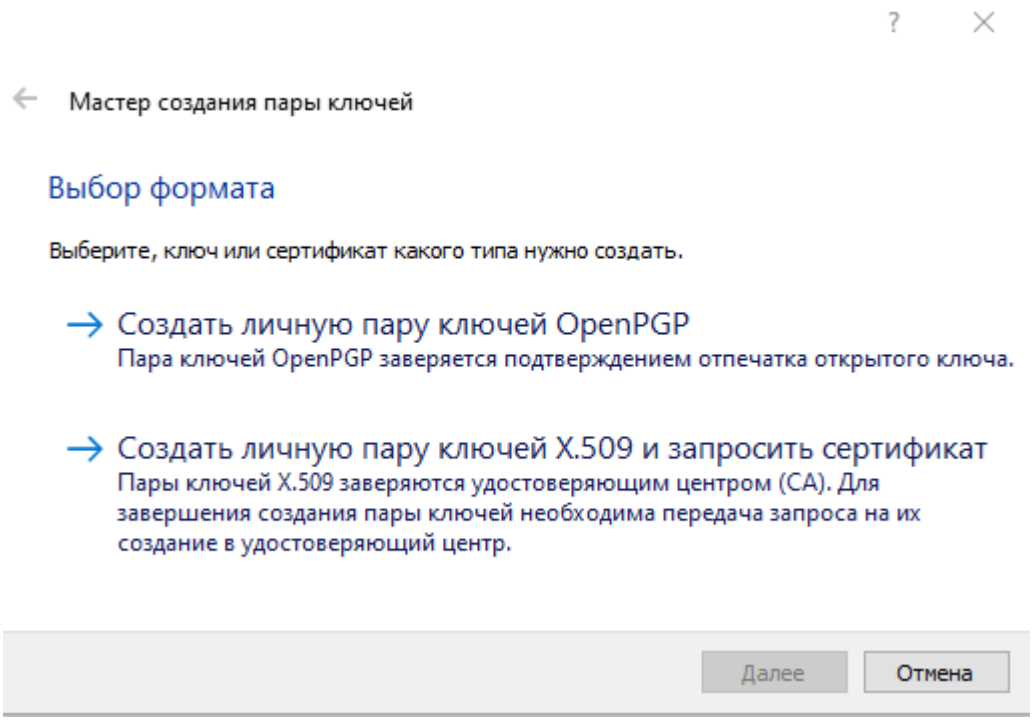

Рисунок 5.2. Процедура создания ключевой пары.

В данной практической работе следует продемонстрировать работу программы по стандарту OpenPGP, поэтому необходимо нажать верхнюю кнопку и перейти в окно, где потребуется заполнить информацию о владельце ключа.

Выбрать алгоритм цифровой подписи, длину ключа и время его действия (Рисунок 5.4).

Проверить правильность введенных данных в следующем окне. На этом процедура генерации ключей завершена. Информацию о созданных ключах можно найти в основном меню программы.

Процедура экспорта и импорта ключей

Для обмена открытыми частями ключевых пар необходимо иметь возможность экспортировать и импортировать открытые части ключей. Для экспорта открытого ключа необходимо выбрать нужный сертификат из списка главного окна программы Kleopatra и выбрать пункт меню ***File → Export Certificates***.

Сохранить файл с расширением ***.asc.*** Для импорта чужого открытого ключа необходимо выбрать пункт меню File →Import Certificates и выбрать файл сертификата (в данной практической работе операции экспорта и импорта производятся с файлами формата ***.asc***).

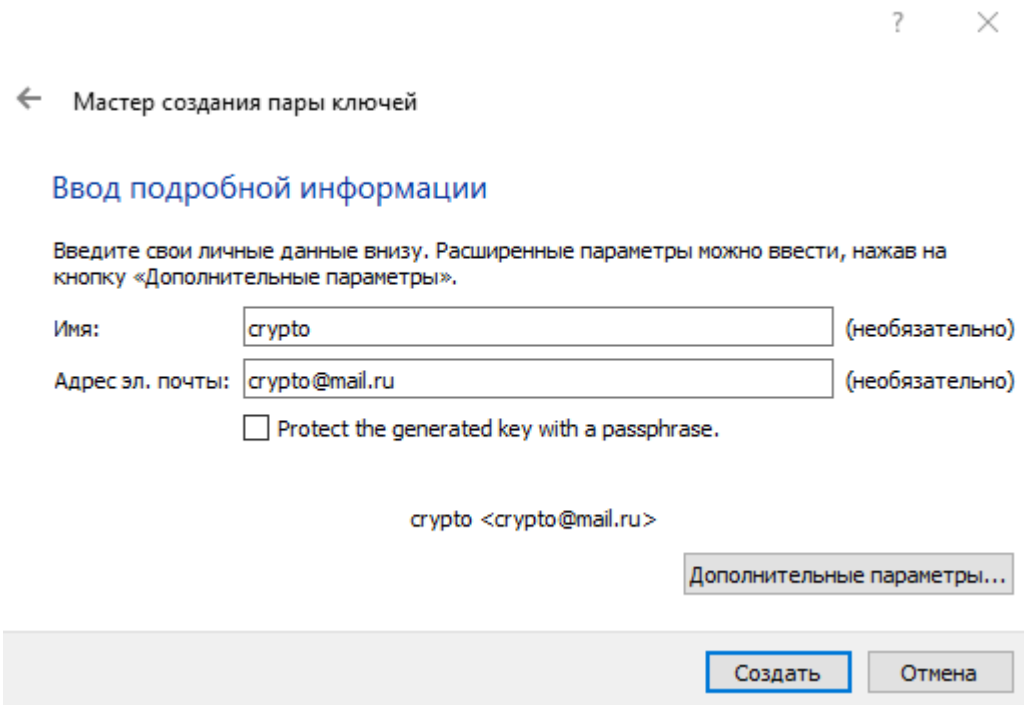

Рисунок 5.3. Привязка ключевой пары к аккаунту

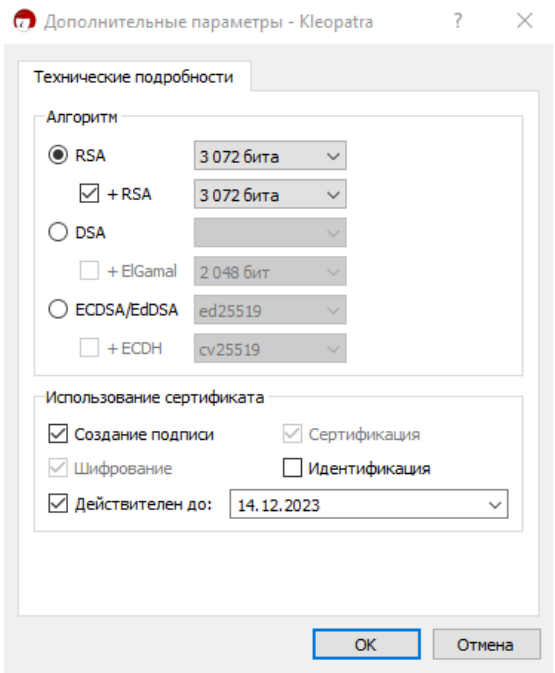

Рисунок 5.4. Настройка параметров алгоритма подписи

Шифрование и цифровая подпись файлов

Процедуры шифрования и цифровой подписи файлов в менеджере ключей

Kleopatra проходят схожим образом. В Kleopatra выбирается пункт меню ***File*** → ***Sign/Encrypt Files*** и выбирается целевой файл, после чего открывается окно (Рисунок 5.5).

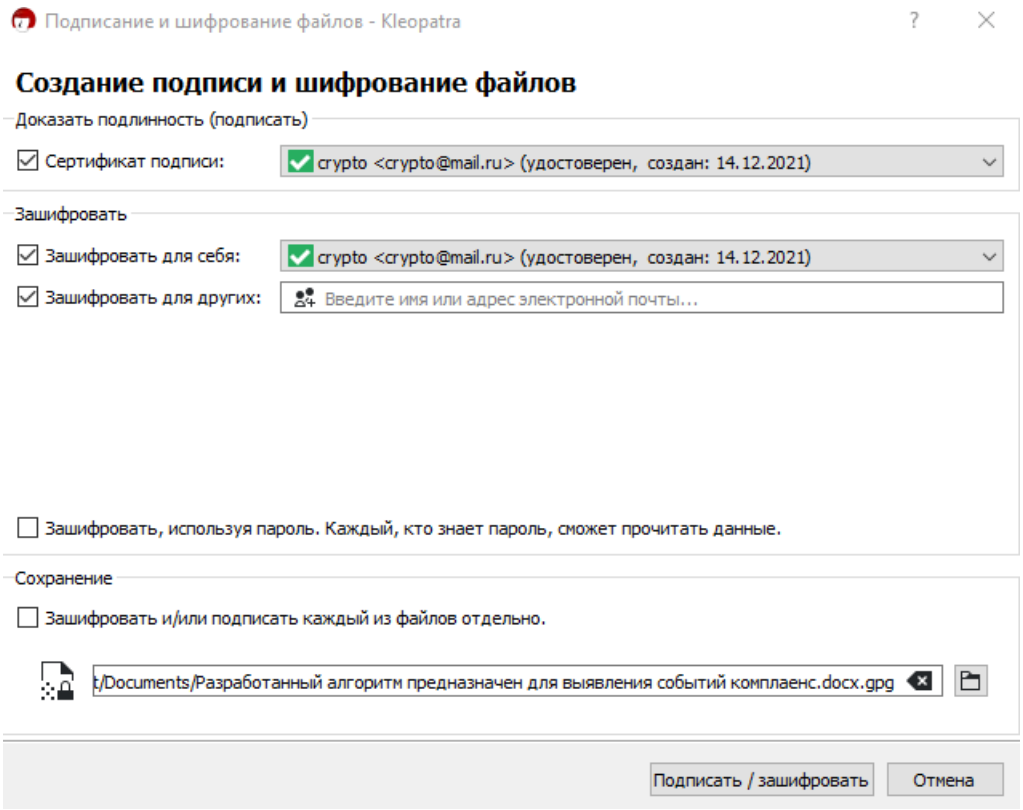

Рисунок 5.5. Меню создания подписи и шифрования файлов.

В данном окне можно сделать выбор, какую процедуру осуществить с файлом: Encrypt (зашифровать) и Sign (подписать).

В данном окне необходимо выбрать сертификаты тех, для кого предназначено сообщение. Здесь можно задать процесс шифрования таким образом, чтобы сообщение могли расшифровать несколько человек. После нажатия кнопки ***Encrypt*** происходит процесс шифрования.

В результате будет создан файл формата ***.gpg***, который является зашифрованным файлом. Для создания цифровой подписи необходимо подтвердить использование стандарта OpenPGP. Далее ввести пароль для закрытой части ключевой пары, если он указывался при ее создании. В результате будет создан файл формата ***.sig***, который будет являться цифровой подписью файла (Рисунок 5.6).

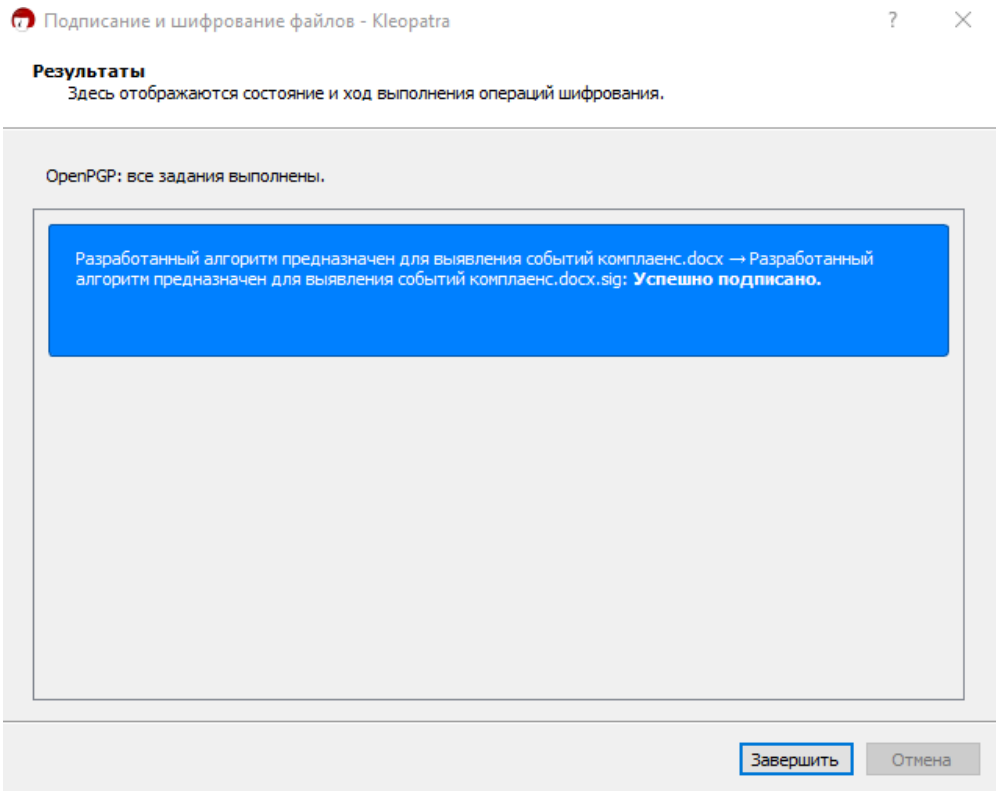

Рисунок 5.6. Генерация цифровой подписи.

Процедура дешифрования и проверки цифровой подписи

Процедуры дешифрования и проверки цифровой подписи файлов в менеджере ключей Kleopatra проходят схожим образом. В Kleopatra выбирается пункт меню ***File → Decrypt/Verify Files*** и выбирается зашифрованный файл (.gpg) или цифровая подпись (.sig).

В результате будет получен дешифрованный файл или сообщение об успешной проверке цифровой подписи (Рисунок 5.7).

Порядок выполнения работы

1. Установить GnuPGP и менеджер ключей Kleopatra на свою операционную систему.
2. Сгенерировать новую пару ключей (создать новый сертификат), следуя инструкциям, данным в вышеприведенных частях данной практической работы.
3. Экспортировать открытую часть сгенерированной пары ключей в файл ***key.asc*** и приложить к отчету.
4. Составить небольшой файл с названием ***notion.doc***, содержащий краткое определение термина (3-4 предложения), в зависимости от выбранного Вами варианта из Таблицы 1.

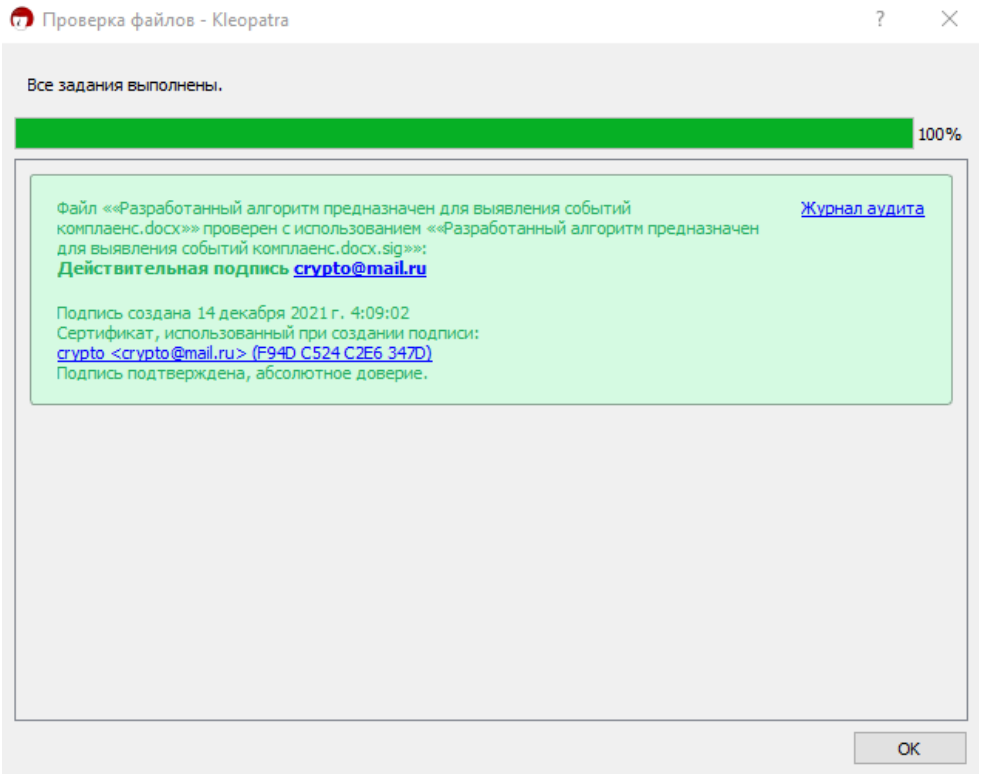

Рисунок 5.7. Сообщение об успешной проверке цифровой подписи.

4. Создать цифровую подпись для файла ***notion.doc***, используя сгенерированную пару ключей, и приложить файл цифровой подписи ***notion.doc.sig*** к отчету.
5. Осуществить проверку созданной цифровой подписи и отразить результат в отчете.
6. Зашифровать файл ***notion.doc***, используя импортированный открытый ключ (файл ***crypto.asc***), который находится в приложении к тексту данной практической работы, и приложить к отчету результат шифрования ***notion.doc.gpg.***

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;
- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

Таблица 1. Термины для шифрования данных и генерации подписей

Вариант	**Термин**
1	Симметричная система шифрования
2	Асимметричная система шифрования
3	Поточный шифр
4	Криптографический протокол
5	Цифровая подпись
6	Хэш
7	Криптостойкость шифра
8	Brute-force
9	Имитозащита
10	Имитовставка
11	Криптоанализ
12	Side channel attack
13	Ключ
14	Блочный шифр
15	Цифровой сертификат
16	Факторизация целых чисел
17	Дискретное логарифмирование

Кроме обязательных элементов отчета (титульный лист, цель, выводы, ход работы со скриншотами, подтверждающими выполнение задач практической работы) необходимо прикрепить в приложение три следующих файла:

- key.asc (экспортированная открытая часть ключа);
- notion.doc.sig (цифровая подпись файла, сделанная с помощью сгенерированной пары ключей);
- notion.doc.gpg (зашифрованный с помощью некоторого импортированного ключа).

6. Модель протокола защищенного соединения

Цель: изучить подходы к применению криптопримитивов в рамках протоколов для защищенных соединений.

Необходимое программное обеспечение

В рамках задания необходим установленный OpenSSL, который может быть установлен отдельно или как составляющая сборки (например, kali linux).

Ход работы

В ходе работы необходимо изучить взаимодействие таких криптопримитивов, как алгоритмы симметричного и асимметричного шифрования, имитовставки и электронной подписи. В рамках практического задания предлагается рассмотреть один из способов, как эти криптопримитивы могут взаимодействовать между собой.

1. Выполнить визуализацию 1 раунда любого алгоритма хэширования Templates->Hash Functions->MD5 (MD5, SHA-1, SHA-2, SHA-3) или имитовставки (Рисунок 6.1). Начальные переменные, раундовые функции определяются согласно стандартам хэширования.

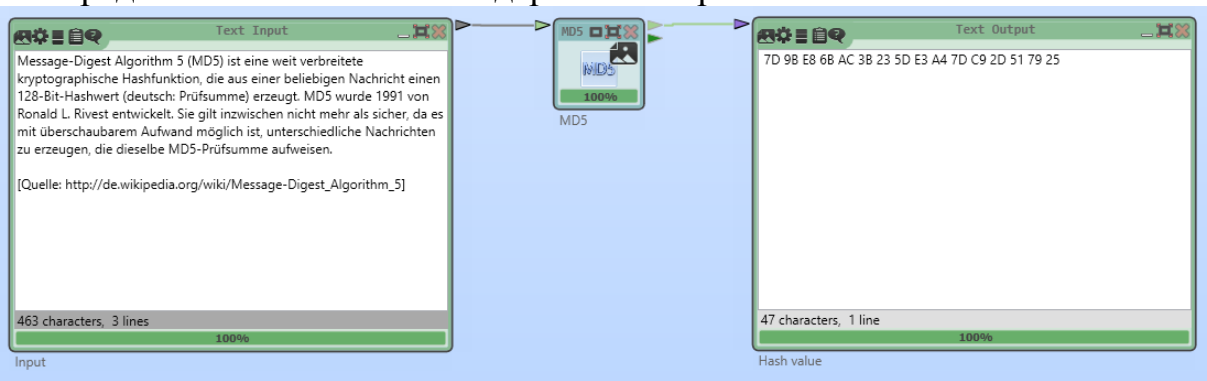

Рисунок 6.1. Модель алгоритма хэширования MD5

Проанализировать каждую из операций, входящую в раундовую функцию для хэширования (Рисунок 6.2). Отдельно описать процесс инициализации раундовых констант A, B, C, D, констант для сложения и сдвига 32 битного слова.

2. Отразить в отчете алгоритм для генерации кодов аутентификации сообщений. Путь для шаблонов Cryptool 2 Templates->Hash Functions->HMAC.
3. Протестировать возможности хэширования файлов и сообщений с помощью openssl dgst. Выполнить хэширование как минимум для одного сообщения (задаваемого, например, через echo) и одного файла.

Протестировать хэш-алгоритмы –md5, -sha256, -sha512. Проанализируйте длину сгенерированного хэша. Влияет ли на длину хэша длина входного сообщения? Как сильно меняется хэш, если мы незначительно изменим входное сообщение (на 1 бит, на 1 символ)?

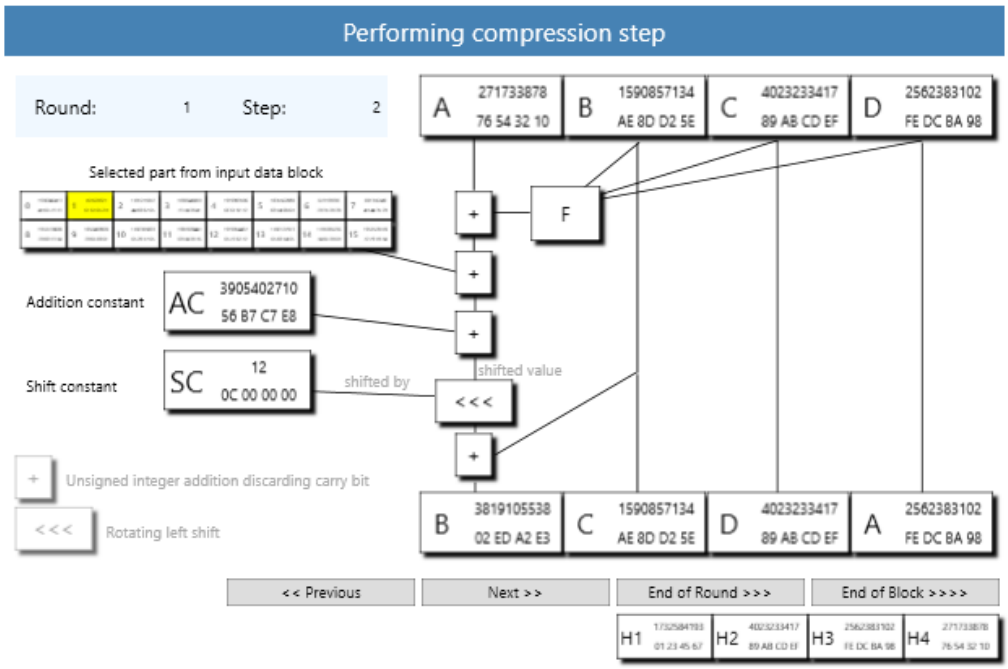

Рисунок 6.2. Структура алгоритма хэширования MD5.

4. Сгенерировать Hash based MAC. Для этого воспользуйтесь дополнительным флагом –hmac. После флага –hmac укажите ключ. Сгенерируйте, используя различные флаги и ключи, HMAC-MD5, HMAC-SHA256, HMAC-sha512. Проверьте возможность использования ключей различной длины при генерации имитовставки. Возможно ли это? Требуется ли указывать ключ фиксированного размера?
5. Сгенерировать пару открытый-закрытый ключ для асимметричной криптосистемы RSA. С помощью данной ключевой пары RSA зашифруйте случайный ключ AES, например, длиной 128 бит. Показать, с помощью каких команд в OpenSSL можно выполнить:
   - Генерацию ключевой пары;
   - Шифрование симметричного ключа с помощью открытого ключа криптосистемы RSA;
   - Дешифрование симметричного ключа;
   - Генерацию подписи для зашифрованного сообщения на основе асимметричного алгоритма RSA;
   - Проверку подписи RSA.
6. Поскольку для алгоритма симметричного шифрования необходим ключ, то мы должны передать его принимающей стороне. Для этого мы используем

асимметричные криптоалгоритмы с электронной цифровой подписью в каждом сообщении. Сообщением для асимметричного криптоалгоритма является ключ симметричной криптосистемы, задаваемый на шаге 1. Выполнить генерацию ключей для асимметричного криптоалгоритма (например, RSA или другого), зашифровать сообщение, сгенерировать подпись и выполнить проверку подписи соответствующими командами:

- *openssl genrsa (генерация ключей)*;
- *openssl rsautl –encrypt (шифрование);*
- *openssl dgst и openssl rsautl -sign (для генерации хэш значения, взятого от симметричного ключа, и подписи сгенерированного хэша);*
- *openssl rsautl –verify (для проверки подписи)*.

7. В качестве сообщения может быть указана случайная строка (задаваемая через echo) или файл (example.txt). Продемонстрировать с помощью команд OpenSSL процесс шифрования и дешифрования сообщения с помощью любой симметричной криптосистемы, используя команду **openssl enc** с любыми флагами (т.е. сам алгоритм шифрования, режим шифрования, способ задания ключа и прочие настройки выбрать самостоятельно из набора предоставляемого OpenSSL).
8. Дополнительно с зашифрованным сообщением использовать алгоритм имитовставки с тем же ключом шифрования, который использовался при шифровании сообщения. Сгенерировать hmac для сообщения из пункта 1. Научиться сравнивать между собой два разных hmac с помощью флага *-verify* или команды *diff* .

Требования к отчету

Отчет о выполнении практического задания должен включать:

- титульный лист с подписью студента;
- цель работы и выводы;
- описание выполненных команд и протокол действий студента (с подтверждениями в виде скриншотов результатов работы программы);
- входные и выходные данные криптосистем, а также информацию о криптографическом ключе;
- наблюдения студента о процессах шифрования, дешифрования, уязвимостях рассматриваемых в работе криптосистем.

Отчеты загружаются в электронном виде в каталог, доступ к которому предоставляется преподавателем.

Таранов Сергей Владимирович

КРИПТОГРАФИЧЕСКИЕ ПРОТОКОЛЫ И СТАНДАРТЫ. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

Учебно-методическое пособие

В авторской редакции
Редакционно-издательский отдел Университета ИТМО
Зав. РИО Н.Ф. Гусарова
Подписано к печати
Заказ №
Тираж
Отпечатано на ризографе

Редакционно-издательский отдел
Университета ИТМО
197101, Санкт-Петербург, Кронверкский пр., 49, литер А