**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационной безопасности**

отчет

**по практической работе №8**

**по дисциплине «Криптографические методы защиты информации»**

**Тема: «Изучение электронной подписи»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9361 |  | Кисляков Н. |
| Преподаватель |  | Племянников A.К. |

Санкт-Петербург

2023

**Цель работы**

Исследовать алгоритмы создания и проверки электронной подписи, алгоритмы генерации ключевых пар для алгоритмов электронной подписи RSA, DSA, ECDSA и получить практические навыки работы с ними, в том числе с использованием приложения CrypTool версий 1 и 2.

1. **Генерация ключевых пар**
   1. **Задание**

1. Перейти к утилите «Digital Signatures/PKI –> PKI/Generate…».

2. Сгенерировать ключевые пары по алгоритмам RSA-2048, DSA-2048, EC-239. Зафиксировать время генерации в таблице.

3. С помощью утилиты «Digital Signatures/PKI –> PKI/Display…» вывести сгенерированный открытый ключ и сохранить соответствующий скриншот.

* 1. **Описание алгоритмов генерации ключевых пар**

*Генерация ключевых пар для алгоритма RSA:*

1. Выбираются два больших простых числа 𝑝 и 𝑞;

2. Вычисляется 𝑛 = 𝑝 ∙ 𝑞;

3. Выбирается произвольное число 𝑒 (𝑒 < 𝑛), взаимно простое с 𝜑(𝑛) (𝜑(𝑛) = (𝑝 − 1) ∙ (𝑞 − 1));

4. Вычисляется ;

5. Пара чисел (𝑒, 𝑛) объявляются открытым ключом, а 𝑑 – закрытым ключом.

6. Числа 𝑝 и 𝑞 уничтожаются.

*Генерация ключевых пар для алгоритма DSA:*

1. Выбирается простое число 𝑝𝑝, длиной между 512 и 1024 битами, число битов в 𝑝 должно быть кратно 64;

2. Выбирается другое простое число 𝑞, которое имеет тот же самый

размер, что и дайджест – 160 битов, такое, что ;

3. Выбирается , такое, что , путем вычисления (теорема Ферма);

4. Выбирается целое и вычисляется ;

5. Числа () объявляются открытым ключом, а 𝑑 – закрытым ключом.

*Генерация ключевых пар для алгоритма ECDSA:*

1. Выбирается эллиптическая кривая , 𝑝 – простое число;

2. Выбирается точка на эллиптической кривой ;

3. Для дальнейших вычислений выбирается другое простое число 𝑞 – порядок циклической подгруппы группы точек эллиптической кривой: ;

4. Выбирается целое число 𝑑, и назначается закрытым ключом;

5. Вычисляется другая точка на кривой ;

6. Числа () объявляются открытым ключом.

* 1. **Таблица с фактическими временем генерации ключевых пар**

Используем в Cryptool 1 утилиту «Digital Signatures/PKI -> PKI/Generate…» для генерации ключевых пар по алгоритмам RSA-2048, DSA-2048, EC-239. В таблице 1 кажем время, которое тратится для генерации ключа.

*Таблица 1*

|  |  |
| --- | --- |
| Алгоритм | Время генерации ключевых пар |
| RSA-2048 | 1,721 |
| DSA-2048 | 1,207 |
| EC-239 (prime239v1) | 0,014 |

Как видно из таблицы 1, EC-239 занимает минимальное количество времени для генерации ключей.

* 1. **Скриншоты со значениями открытых ключей**

На рисунке 1, 2, 3 представлены сгенерированные открытые ключи. Которые мы получили в прошлой под главой.

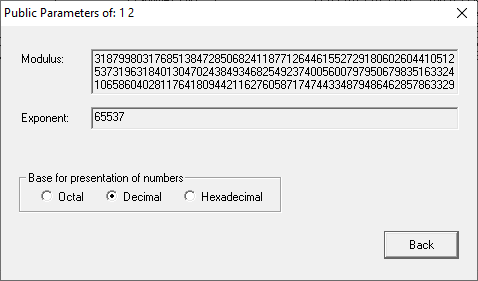


Рисунок 1 – Открытый ключ RSA-2048

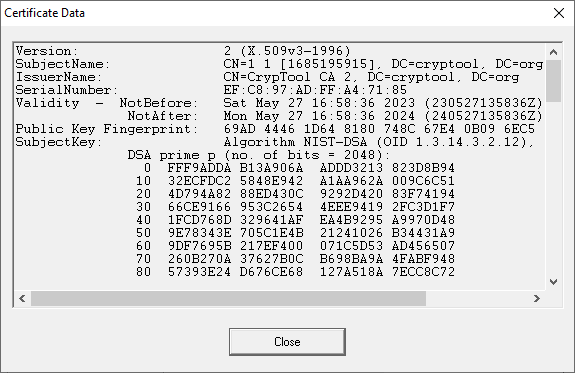


Рисунок 2 – Открытый ключ DSA-2048

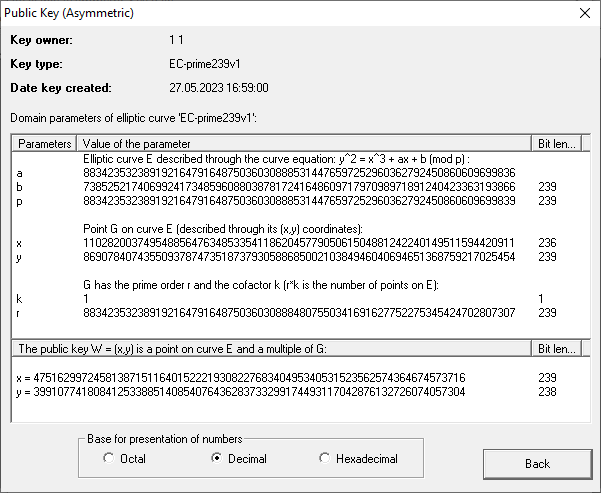


Рисунок 3 – Открытый ключ EC-239

1. **Процессы создания и проверки электронной подписи**
   1. **Задание**

1. Открыть текст не менее 5000 знаков. Перейти к приложению Digital Signatures/PKI –> Sign Document…

2. Задать хеш-функцию и другие параметры электронной подписи.

3. Создать подписи, используя закрытые ключи, сгенерированные в предыдущем задании. Зафиксировать время создания электронной подписи для каждого ключа (опция Display signature time должна быть включена)

4. Сохранить скриншот любой электронной подписи с помощью приложения Digital Signatures/PKI –> Extract Signature.

5. Выполнить процедуру проверки любой подписи Digital Signatures/ PKI –> Verify Signature для случаев сохранения и нарушения целостности исходного текста. Сохранить скриншоты результатов.

* 1. **Обобщенная схема создания и проверки электронной подписи**

Электронная подпись – это некоторая информация в электронной форме (код), которая присоединена к другой информации (файлу данных) с целью подтверждения авторства и контроля целостности файла данных. Обобщенная схема, поясняющая работу протокола подписания документа и проверки электронной подписи, продемонстрирована на рисунке 4.

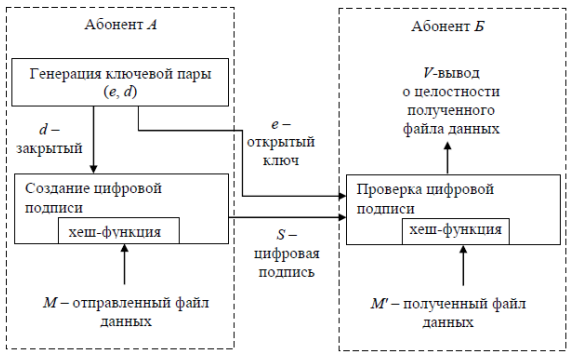


Рисунок 4 – Обобщенная схема создания и проверки электронной подписи

Следует отметить, что в самом общем случае на стороне отправителя запускается процедура создания электронной подписи (процедура подписания), а на стороне получателя – процедура проверки электронной подписи (процедура верификации). Подпись создается на основе хэш-кода отправляемого файла данных и закрытого ключа отправителя, а при проверке подписи используется хеш-код от полученного файла данных, сопровождающая его подпись и открытый ключ отправителя, доставленный получателю.

* 1. **Таблица с фактическими временами создания электронной подписи различными алгоритмами**

На рисунке 5 представлен исходный текст, который состоит из 143265 знаков с пробелами.

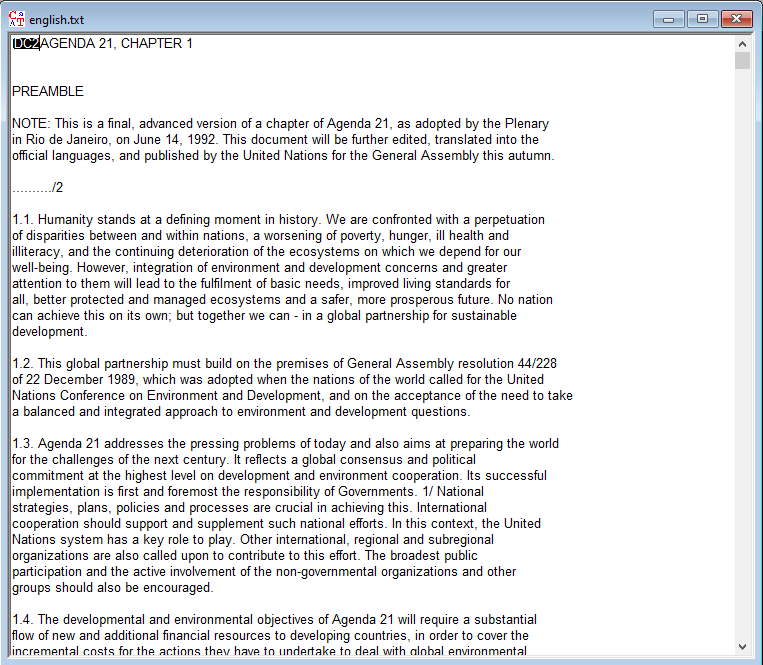
****

Рисунок 5 – Исходный текст

С помощью сгенерированных ключевых пар, создадим цифровые подписи и зафиксируем время генерации в таблице 2.

*Таблица 2*

|  |  |  |
| --- | --- | --- |
| Алгоритм | Хэш-функция | Время генерации цифровой подписи |
| RSA | SHA-1 | 0,012 секунды |
| DSA | SHA-1 | 0,004 секунды |
| ECSP-DSA | SHA-1 | 0,004 секунды |
| ECSP-NR | SHA-1 | 0,004 секунды |

* 1. **Скриншот со значением электронной подписи**

Сгенерированная цифровая подпись алгоритмом RSA и хэш-функцией SHA-1 представлена на рисунке 6.

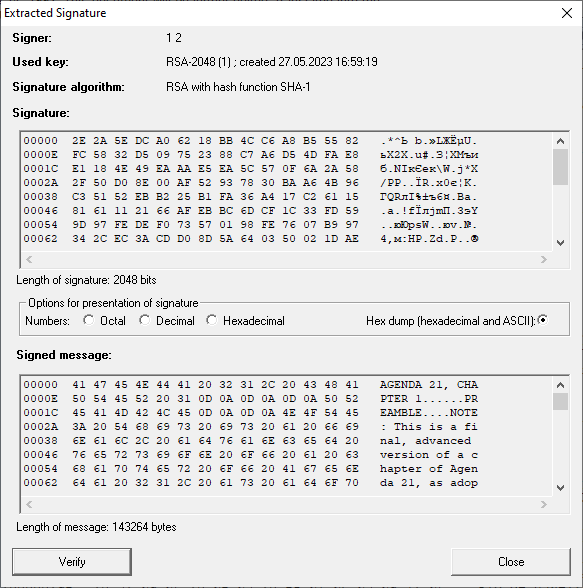


Рисунок 6 – Цифровая подпись

* 1. **Скриншоты с результатами проверки электронной подписи**

Выполним процедуру проверки цифровой подписи, сгенерированной алгоритмом RSA и хэш-функцией SHA-1, для случаев сохранения и нарушения целостности исходного текста. Результаты проверки продемонстрированы на рисунках 7 и 8.

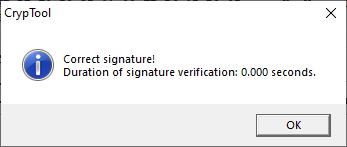


Рисунок 7 – Проверка при сохранении

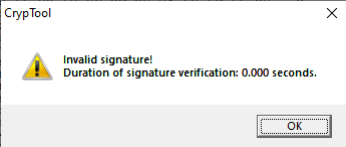


Рисунок 8 – Проверка при нарушении

1. **Создание и проверка электронной подписи на основе эллиптических кривых**
   1. **Задание**

1. Выполнить процедуру создания подписи Digital Signatures/PKI –> Sign Document… алгоритмом ECSP-DSA в пошаговом режиме (Display inter. results = ON). Зафиксировать скриншоты последовательности шагов.

2. Выполнить процедуру проверки подписи ECSP-DSA для случаев сохранения и нарушения целостности исходного текста. Сохранить скриншоты результатов.

3. Проверить лекционный материал по ECDSA, создав и проверив подпись сообщения M (принять M = h(M)) приложением Indiv.Procedures –> Number Theory… –> Point Addition on EC.

**3.2 Описание алгоритма формирования и проверки подписи ECDSA**

Схема цифровой подписи ECDSA показана на рисунке 9.

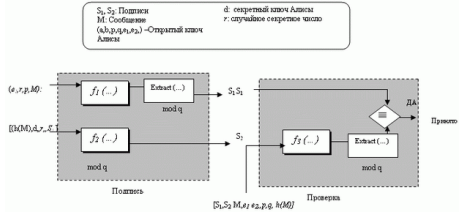


Рисунок 9 – Схема цифровой подписи ECDSA

Функция создает новую точку для секретного ключа подписывающего лица. Функция создает новую точку из двух общедоступных ключей подписывающего лица. Каждый экстрактор извлекает первые координаты соответствующей точки в модульной арифметике.

В процессе подписания две функции и и экстрактор (извлекающее устройство) создают две части подписи. В процессе проверки (верификации) обрабатывают выход одной функции (после прохождения через экстрактор) и сравнивают ее с первой частью подписи.

С использование ключевой пары (закрытый ключ – 𝑑, и открытый ключ – ()) осуществляется подписание документа, а затем на принимающей стороне – верификация подписи.

Алгоритм создания электронной подписи ECDSA состоит из следующих операций:

1. Выбирается секретное случайное число 𝑟, ;

2. Выбирается третья точка на кривой ;

3. Используем абсциссу 𝑢𝑢, чтобы вычислить первую часть подписи ;

4. Используем дайджест сообщения ℎ(𝑀), закрытый ключ 𝑑, секретное случайное число чтобы вычислить вторую часть подписи .

Алгоритм проверки цифровой подписи ECDSA включает следующие операции:

1. Используем для получения промежуточных результатов 𝐴 и 𝐵:;

2. Затем восстанавливаем третью точку ;

3. Верификатор сравниваем.

Схема протокола ECDSA представлена на рисунке 10.

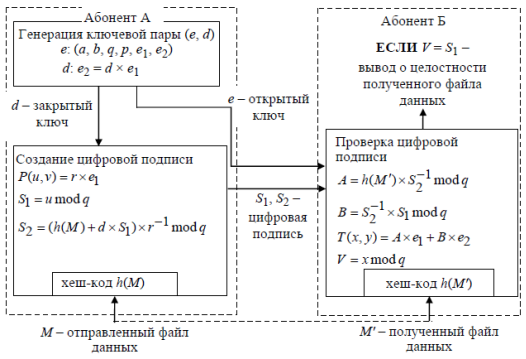


Рисунок 10 – Схема протокола ECDSA

* 1. **Результаты (скриншоты) пошагового выполнения ECDSA в CrypTool 1. Сравнение лекционной версии и реализации**

На рисунке 11, 12, 13, 14, 15, 16 и 17 изображена процедура создания подписи алгоритмом ECSP-DSA в пошаговом режиме при помощи CrypTool 1.

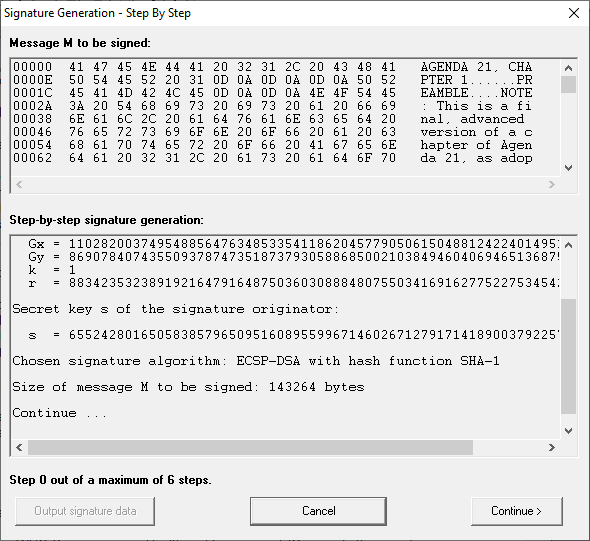


Рисунок 11 – Нулевой шаг создания подписи

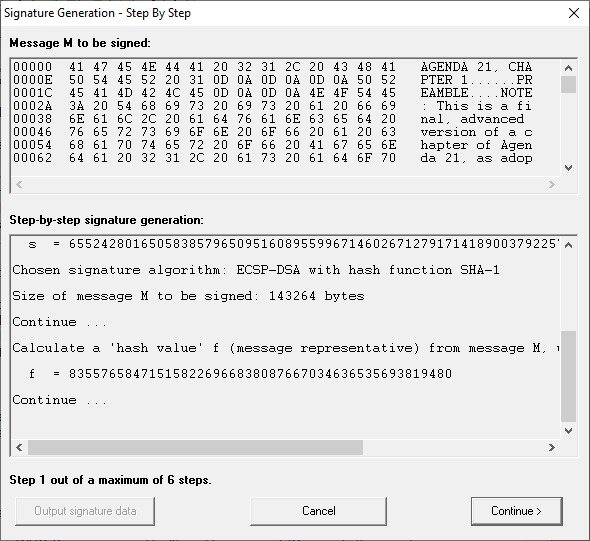


Рисунок 12 – Первый шаг создания подписи

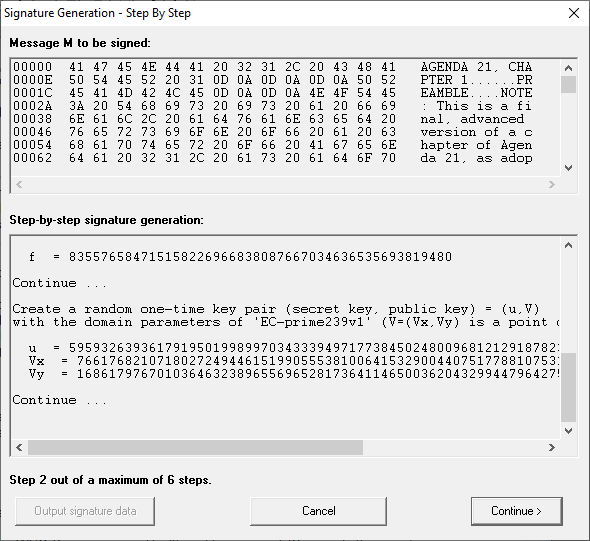


Рисунок 13 – Второй шаг создания подписи

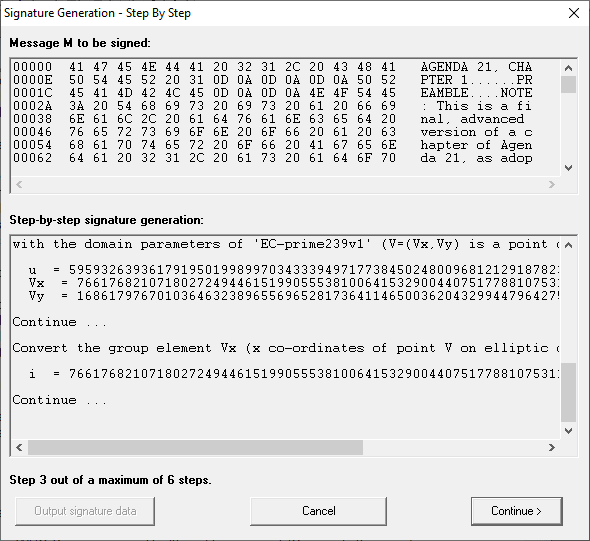


Рисунок 14 – Третий шаг создания подписи

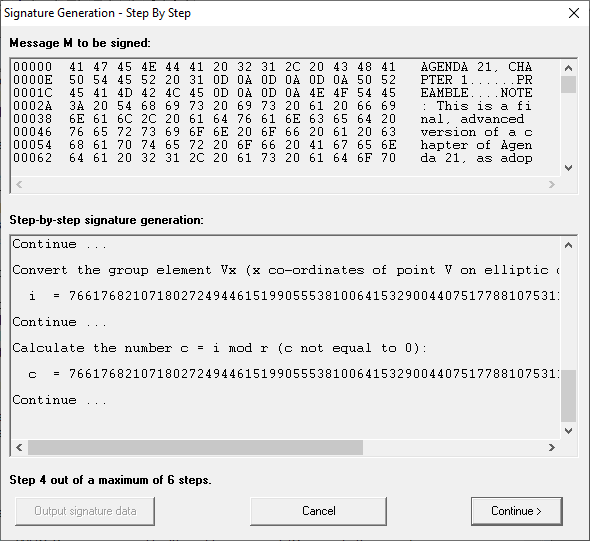


Рисунок 15 – Четвертый шаг создания подписи

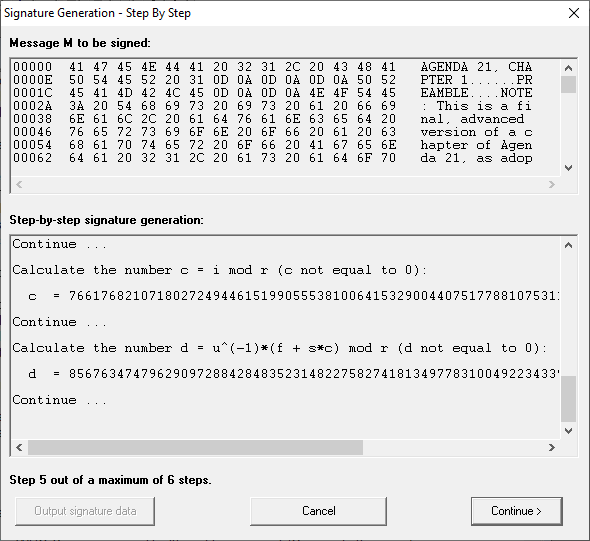


Рисунок 16 – Пятый шаг создания подписи

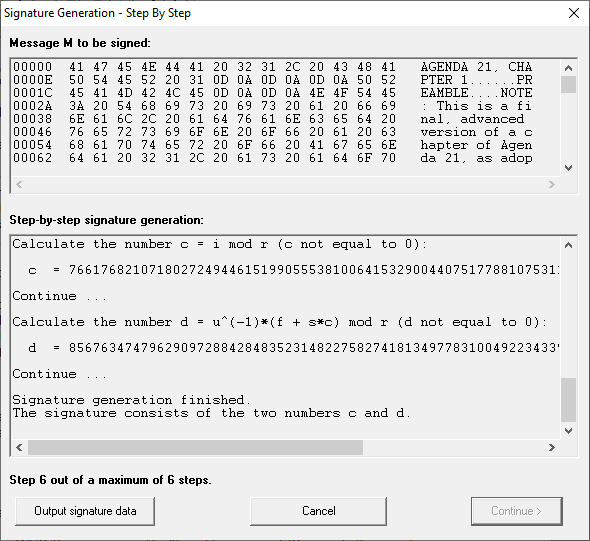


Рисунок 17 – Шестой шаг создания подписи

Теперь сравним лекционную версию ECDSA и реализацию в CrypTool 1. Результаты сравнения представлены в таблице 3.

Таблица 3

|  |  |
| --- | --- |
| Параметр из лекции | Параметр из CrypTool 1 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Выполним процедуру проверки подписи ECSP-DSA для случаев сохранения и нарушения целостности исходного текста. Результаты проверок показаны на рисунках 18 и 19.

****

Рисунок 18 – Проверка цифровой подписи



Рисунок 19 – Проверка цифровой подписи

**3.4 Результаты проверки лекционного материала по ECDSA с использованием приложения Indiv.Procedures –> Number Theory… –> Point Addition on EC**

Выполним проверку лекционного материала по ECDSA с помощью утилиты из CrypTool 1.

1. Генерация ключей ECDSA.

Возьмем следующую эллиптическую кривую: . Далее выберем точку , как представлено на рисунке 20.

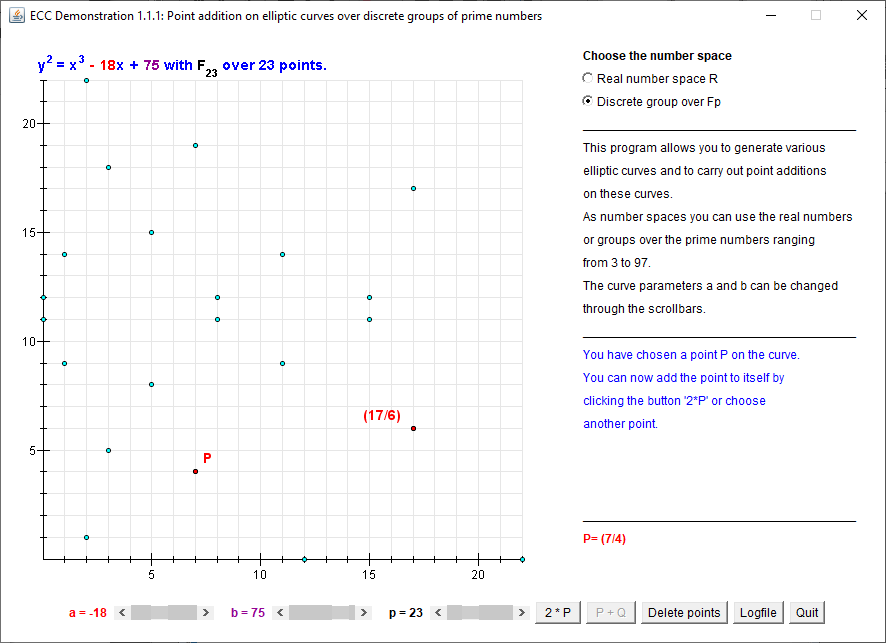
****

Рисунок 20 – Выбор точки

Теперь подберем такое число 𝑞, что . Получилось, что 𝑞 = 11, как продемонстрировано на рисунке 21.

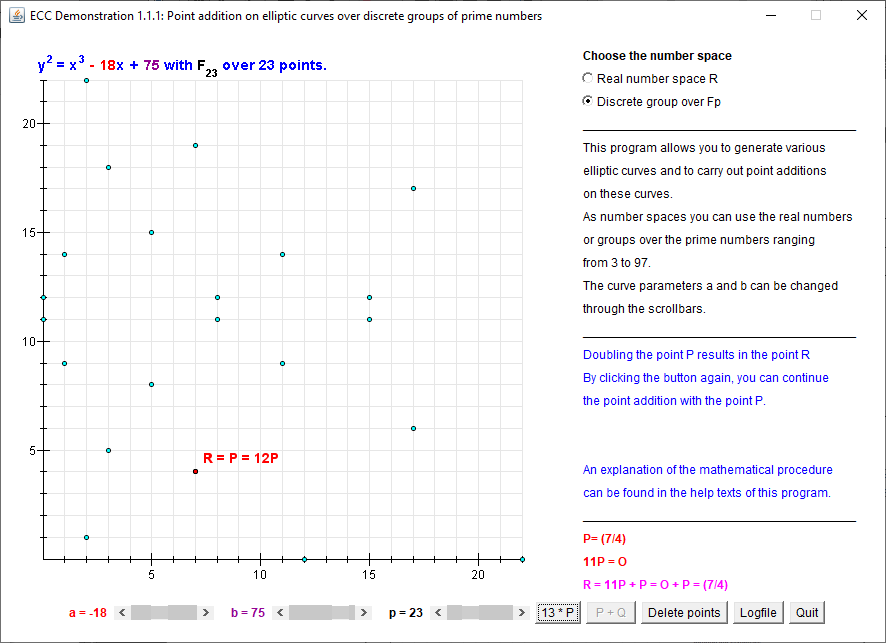


Рисунок 21 – Подбор числа

Выбираем целое число 𝑑𝑑 такое, что 1 < 𝑑 < 𝑞 − 1. Выбрано 𝑑 = 5 – закрытый ключ. После чего, вычисляем точку , как показано на рисунке 22.

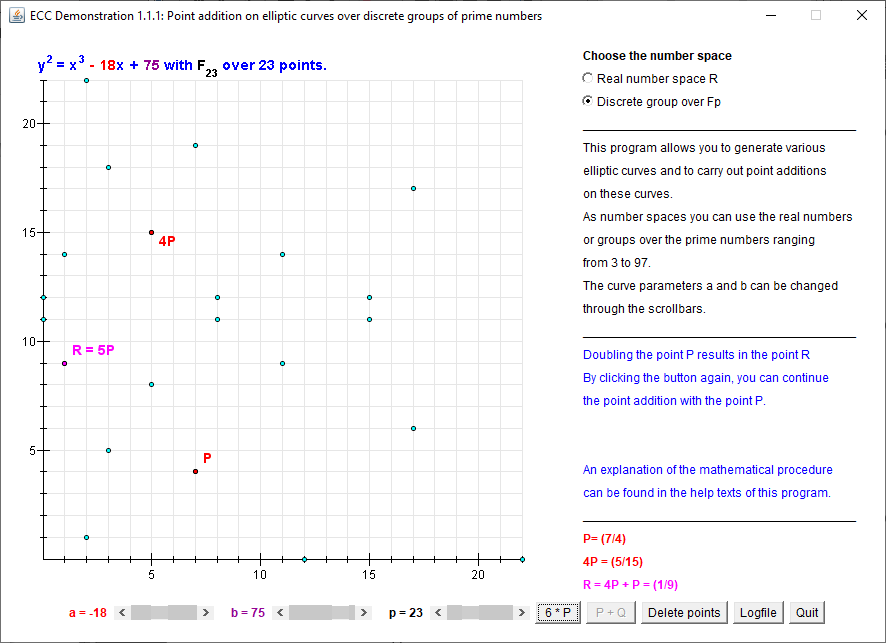


Рисунок 22 – Вычисление точки

Открытый ключ – .

2. ECDSA подписание.

Текст представляется 𝑀 = ℎ(𝑀) = 66. Выбираем случайное 𝑟 = 3.

Находим третью точку .

Вычисляем первую часть подписи: .

Вычисляем вторую часть подписи: .

Тогда получаем .

3. ECDSA проверка.

Вычислим промежуточные результаты 𝐴 и 𝐵:

;

.

Восстанавливаем третью точку 𝑇(𝑥,𝑦): .

Вычисляем верификатор 𝑉: .

Так как следовательно, текст сохранил свою целостность

1. **Демонстрация процесса подписи в среде PK**
   1. **Задание**

1. Запустить демонстрационную утилиту «Digital Signatures/PKI –> Signature Demonstration…».

2. Получить сертификат ключа проверки электронной подписи (открытого ключа) на ранее сгенерированную ключевую пару RSA-2048.

3. Выполнить и сохранить скриншоты всех этапов создания электронной подписи документа.

4. Сохранить скриншот полученного сертификата ключа проверки этой электронной подписи.

**4.2 Описание структуры сертификата из CrypTool 1**

На рисунке 23 представлен сертификат ключа проверки электронной подписи.

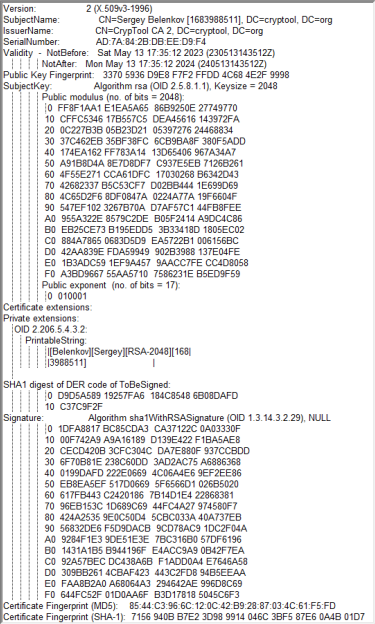
****

Рисунок 23 – Сертификат из CrypTool 1

Как видно из рисунка выше, в структуре сертификата из CrypTool 1 имеется версия сертификата, имя субъекта, имя издателя, серийный номер, период действия, идентификатор алгоритма подписи, информация об открытом ключе (параметры и алгоритм генерации), уникальные идентификаторы издателя и субъекта, дополнительная информация об использовании ключа, а также электронная подпись сертификата.

* 1. **Схема процедуры подписания из CrypTool 1**

Запустим демонстрационную утилиту «Digital Signatures/PKI -> Signature Demonstration…» из CrypTool 1. Выполним все этапы создания цифровой подписи документа. На рисунках 24, 25, 26, 27 и 28 представлены все этапы.

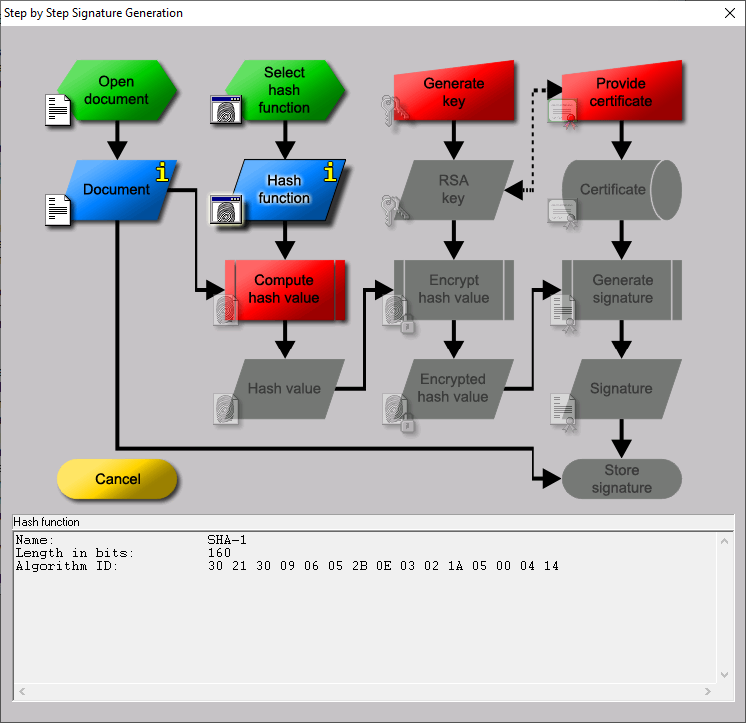
****

Рисунок 24 – Выбор хэш-функции

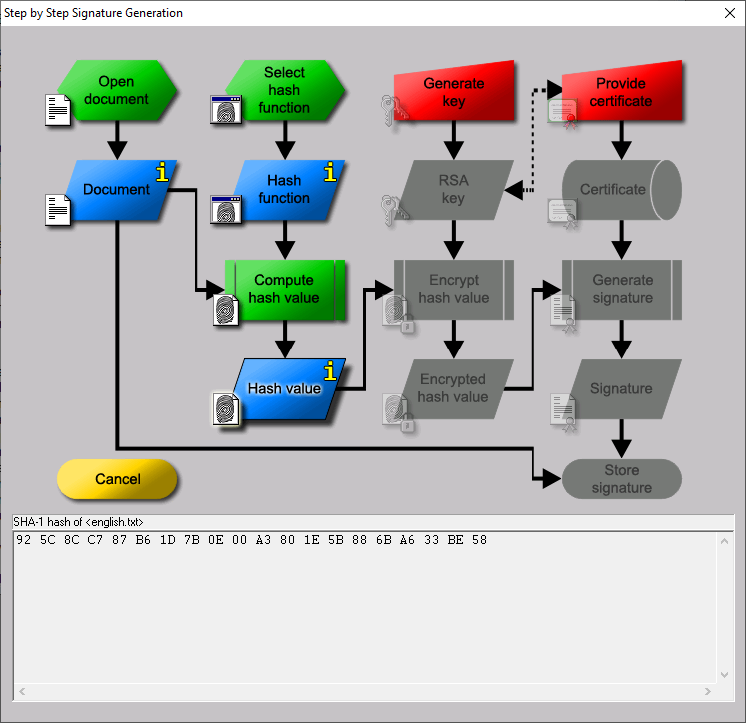
****

Рисунок 25 – Значение хэша документа

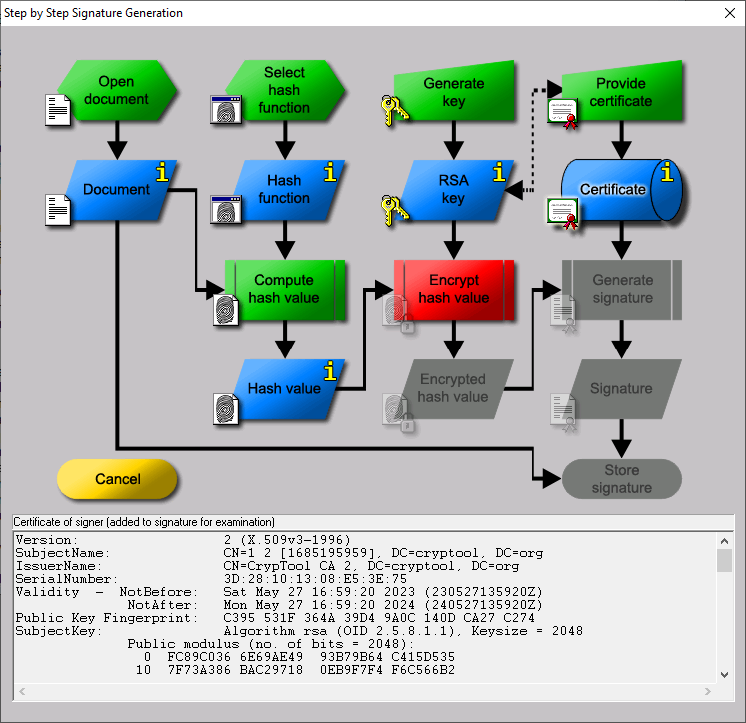
****

Рисунок 26 – Сертификат на сгенерированную ключевую пару

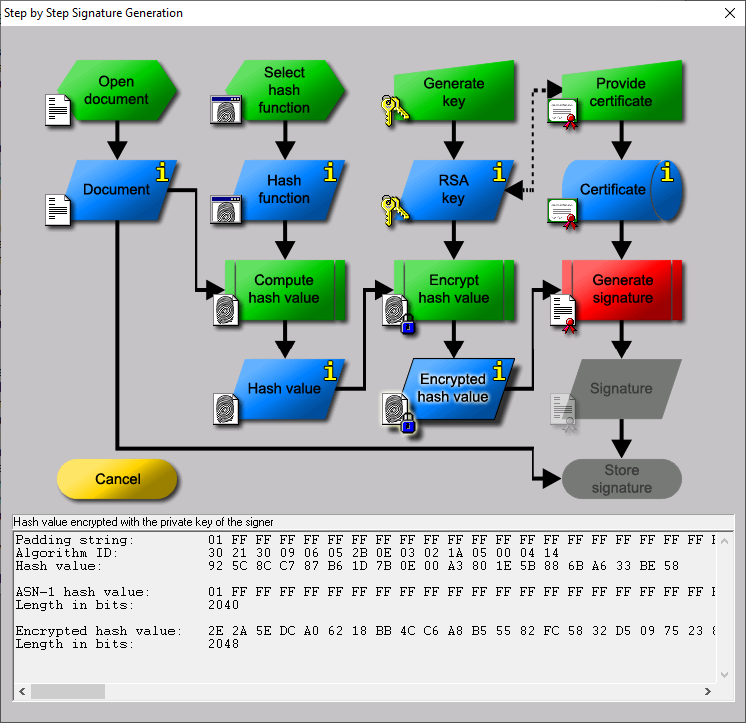
****

Рисунок 27 – Шифрование значения хэша документа

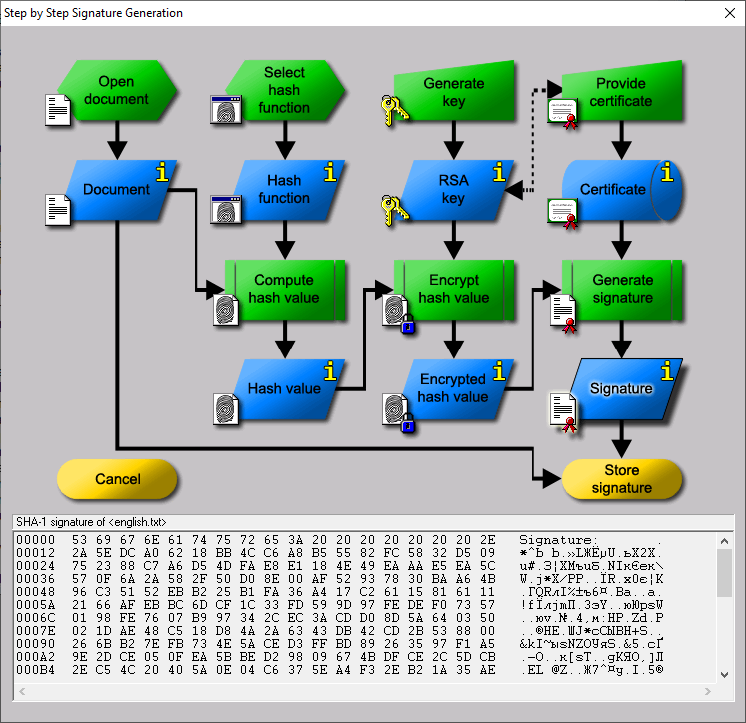
****

Рисунок 28 – Электронная подпись документа

1. **Подписание своего отчета**
   1. **Задание**

1. Сконвертировать отчет в формат pdf.

2. Экспортировать ранее созданный сертификат ключевой пары RSA Digital Signatures/PKI –> PKI/Generate… –> Export PSE(#PKCS12).

3. Открыть pdf-версию отчета и попытаться подписать с использованием этого сертификата.

4. Создать собственный самоподписанный сертификат в среде Adobe Reader и использовать его для подписи отчета.

5. Сохранить скриншоты свойств подписи и сертификата.

6. Внести изменения (маркеры, комментарии) в отчет и проверить подпись.

* 1. **Скриншот титульного листа с электронной подписью**

Для начала текущее состояние отчета было сконвертировано в pdf. Затем при помощи утилиты из CrypTool 1 был экспортирован ранее созданный сертификат ключевой пары RSA. Далее в среде Adobe Reader была осуществлена попытка подписания отчета этим сертификатом, как представлено на рисунке 29.

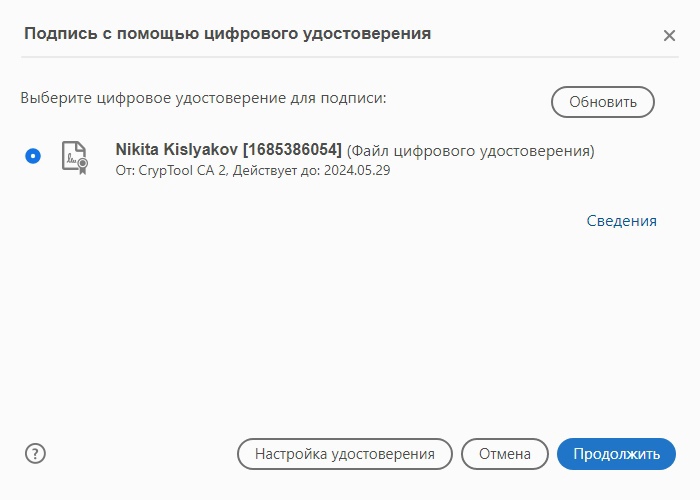


Рисунок 29 – Попытка подписания

В результате подписать pdf-файл не удалось, вылезла ошибка, продемонстрированная на рисунке 30.

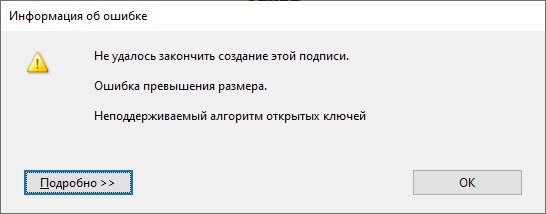


Рисунок 30 – Ошибка при попытке подписать файл

Тогда попробуем создать свой собственный самоподписанный сертификат в среде Adobe Reader и использовать его для подписи отчета. Сертификат показан на рисунке 31.

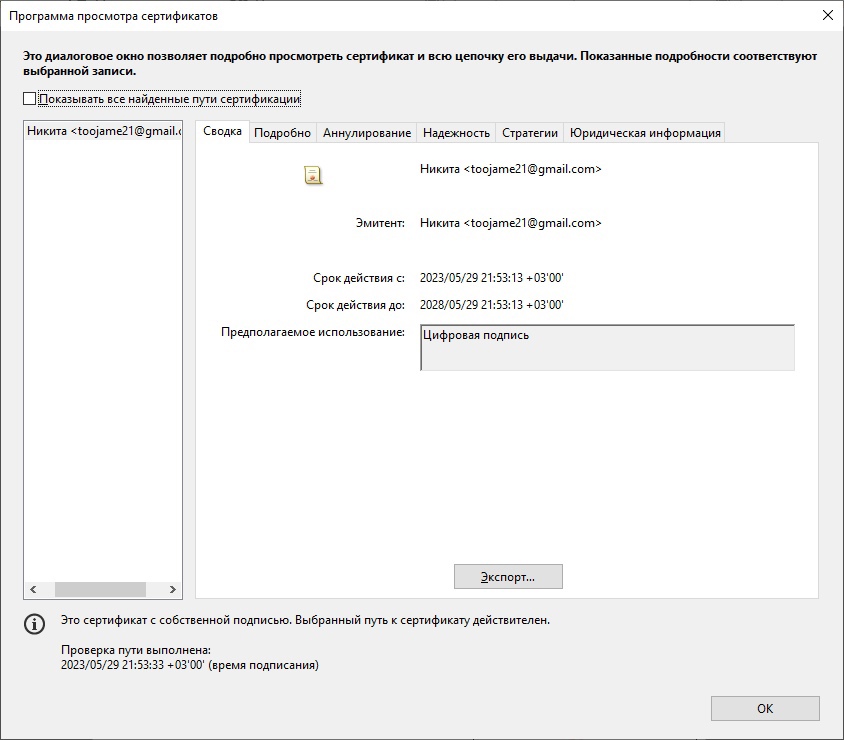


Рисунок 31 – Самоподписанный сертификат

Теперь подпишем наш отчет при помощи созданного сертификата. Титульный лист с электронной подписью представлен на рисунке 32.

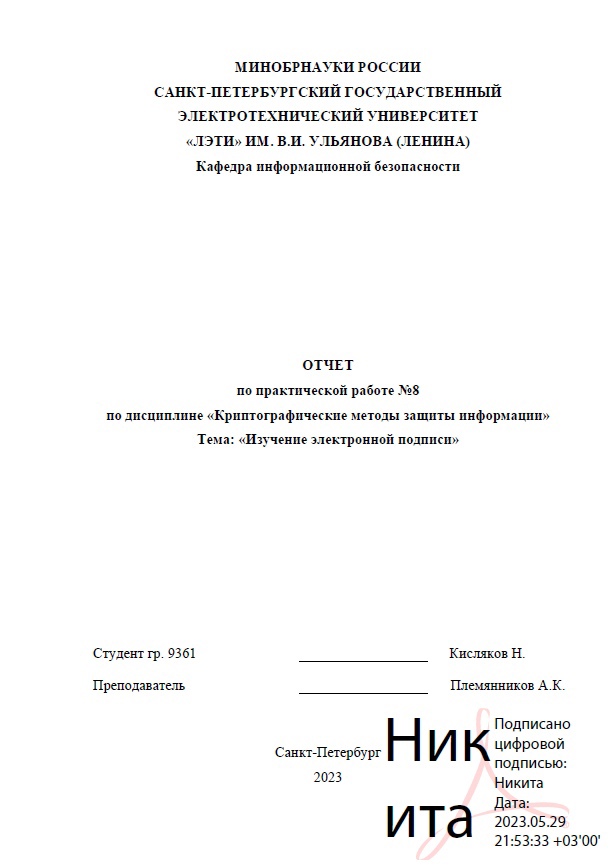


Рисунок 32 – Титульный лист отчета с электронной подписью

* 1. **Скриншоты свойств подписи и сертификата**

Свойства подписи и сертификата продемонстрированы на рисунках 33 и 34.

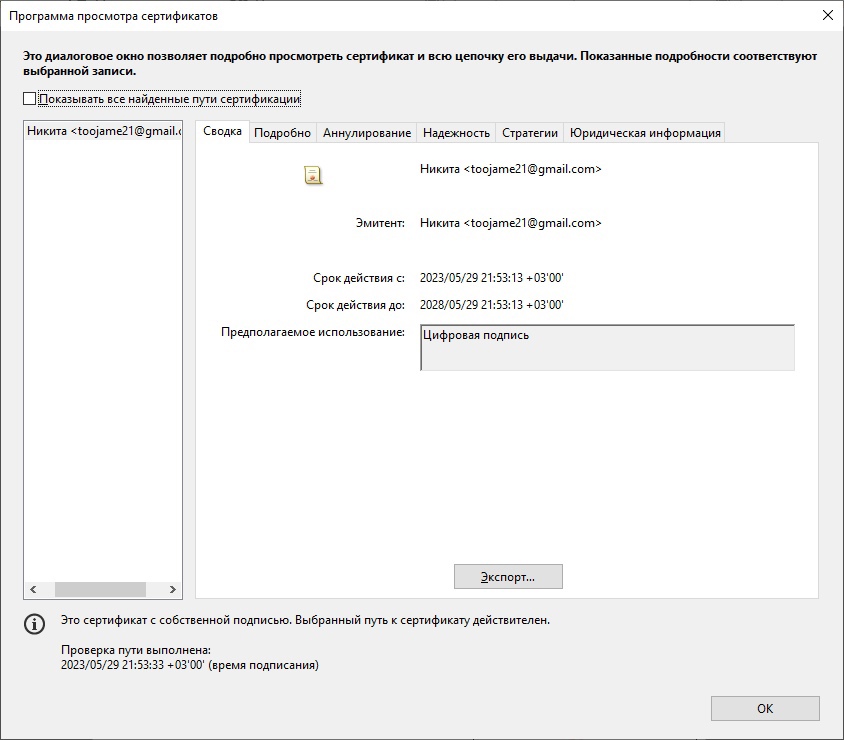


Рисунок 33 – Свойства сертификата

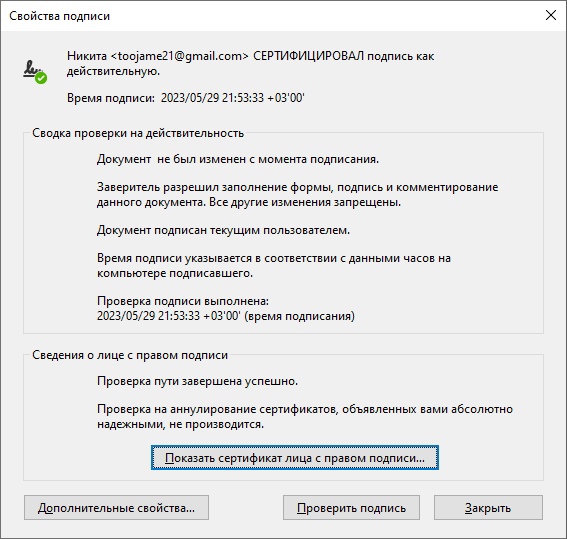


Рисунок 34 – Свойства подписи

* 1. **Скриншот результата проверки после внесения изменений в отчет**

Теперь внесем изменения в pdf-файл и проверим подпись файла.

Результат проверки показан на рисунке 35.

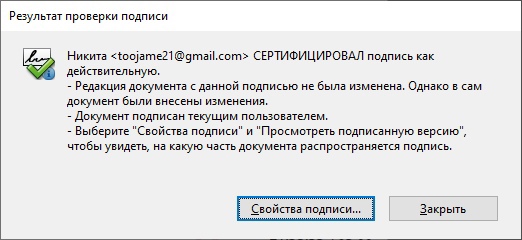


Рисунок 35 – Результат проверки электронной подписи

**Вывод**

В данной лабораторной работе, мы изучили алгоритм и проверку ЭП. Также изучили алгоритм генерации ключевых пар для алгоритмов ЭП RSA, DSA, ECDSA.

1. Генерация ключевых пар.

В данной главе мы изучили работу алгоритмов RSA, DSA, ECDSA. Для каждого алгоритма был описан последовательность действий. Также. С помощью утилиты в CrypTool 1, мы сгенерировали ключевые пары и зафиксировали время генерации ключевых пар.

RSA: было определено, что в качестве открытого ключа алгоритм генерирует пару значений (𝑒, 𝑛), а в качестве закрытого – значение d.

DSA: было определено, что в качестве открытого ключа алгоритм генерирует набор значений (), а в качестве закрытого – значение d.

ECDSA: было определено, что в качестве открытого ключа алгоритм генерирует набор значений (), а в качестве закрытого – значение 𝑑.

Также было измерено время работы рассматриваемых алгоритмов для генерации ключевых пар. Алгоритм EC-239 показал наименьшее время генерации (0,014 секунд), a RSA-2048 – наибольшее (1,721 секунд). Алгоритм DSA-2048 показал время 1,207 секунды

1. Процессы создания и проверки электронной подписи

Была изучена обобщенная схема создания и проверки цифровых подписей. Было установлено, что для создания цифровой подписи требуется вычислить дайджест данных и зашифровать его с помощью закрытого ключа владельца сертификата. Когда цифровая подпись создана, сертификат добавляется к данным вместе с подписью. Для проверки данных можно вычислить дайджест и сравнить его с верификатором, вычисленным путем расшифровки подписи с помощью открытого ключа сертификата.

1. Создание и проверка электронной подписи на основе эллиптических кривых.

Был изучен процесс создания электронной подписи на основе эллиптических кривых. Было определено, что в качестве электронной подписи алгоритм ECDSA генерирует набор значений () при помощи известного открытого ключа – () и закрытого ключа – 𝑑. 𝑀 – это подписанные данные, а – части подписи. Для вычисления и выбирается секретное случайное число 𝑟, лежащее в диапазоне 1 <𝑟 <𝑞 − 1. Далее находится точка на эллиптической кривой . Значение вычисляется как , а значение вычисляется как , где ℎ(𝑀) – дайджест 𝑀.

Был изучен процесс проверки электронной подписи. Было определено, что для проверки электронной подписи необходимо вычислить верификатор 𝑉 и сравнить его со значением . Если их значения совпадают, значит данные не были изменены. Набор значений () получается из электронной подписи и используется для получения промежуточных результатов 𝐴 и 𝐵: . Затем восстанавливается третья точка 𝑇. Далее вычисляется значение верификатора

1. Демонстрация процесса подписи в среде PK

В данном разделе с помощью утилиты CrypTool 1 был изувечен процесс подписи в среде PK. В ходе работы, было обнаружено, что схема создание ЭП в среде PKI совпадает с схемой, которая была рассмотрена раннее. Цифровая подпись содержит зашифрованный дайджест вместе с информацией об алгоритме создания цифровой подписи и данными.

В струтуре CrypTool 1 имеется версия сертификата, имя субъекта, имя издателя, серийный номер, период действия, идентификатор алгоритма подписи, информация об открытом ключе (параметры и алгоритм генерации), уникальные идентификаторы издателя и субъекта, дополнительная информация об использовании ключа, а также

электронная подпись сертификата.

1. Подписание своего отчета.

С помощью Adobe Acrobat Reader мы подписали PDF-файл. Подпись проходила проверку, при сохранении файла, а после изменения PDF-документа – появлялась информация об изменении данных.