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 А.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

1.1 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.

1.2 Описание алгоритмов генерации ключевых пар

Генерация ключевых пар для алгоритма RSA:

1. Выбираются два больших простых числа p и q;

2. Вычисляется $n = p \cdot q$;

3. Выбирается произвольное число e ($e < n$), взаимно простое с $\varphi(n)$ ($\varphi(n) = (p-1) \cdot (q-1)$);

4. Вычисляется d: $e \cdot d = 1 \, mod \, \varphi(n)$;

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 $(p-1) = 0 \, mod \, q$;

3. Выбирается e_1, такое, что $e_1^q = 1\, mod\, p$, путем вычисления $e_1 = e_0^{p-1/q}\ mod\ p$, где $e_0 \in Z_p$ (теорема Ферма);

4. Выбирается целое $d < q$ и вычисляется $e_2 = e_1^d\ mod\ pp$;

5. Числа (e_1, e_2, p, q) объявляются открытым ключом, а d – закрытым ключом.

Генерация ключевых пар для алгоритма ECDSA:

1. Выбирается эллиптическая кривая $E_p(a, b)$, p – простое число;

2. Выбирается точка на эллиптической кривой $e_1 = (x_1, y_1)$;

3. Для дальнейших вычислений выбирается другое простое число q – порядок циклической подгруппы группы точек эллиптической кривой: $q \cdot (x_1, y_1) = O$;

4. Выбирается целое число d, $1 < d < q - 1$ и назначается закрытым ключом;

5. Вычисляется другая точка на кривой $e_2 = d \cdot e_1$;

6. Числа (a, b, p, q, e_1, e_2) объявляются открытым ключом.

1.3 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.

1.4 Скриншоты со значениями открытых ключей

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

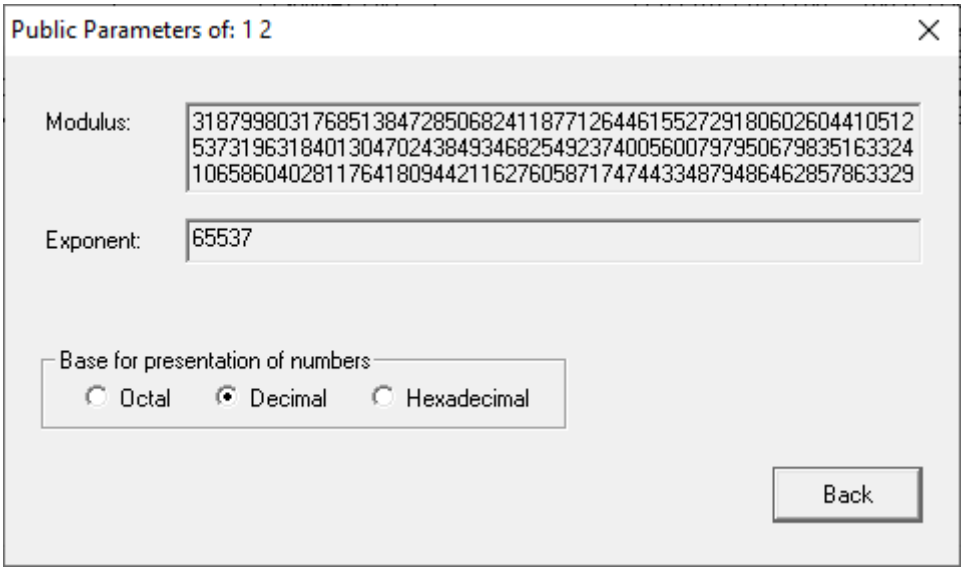


Рисунок 1 – Открытый ключ RSA-2048

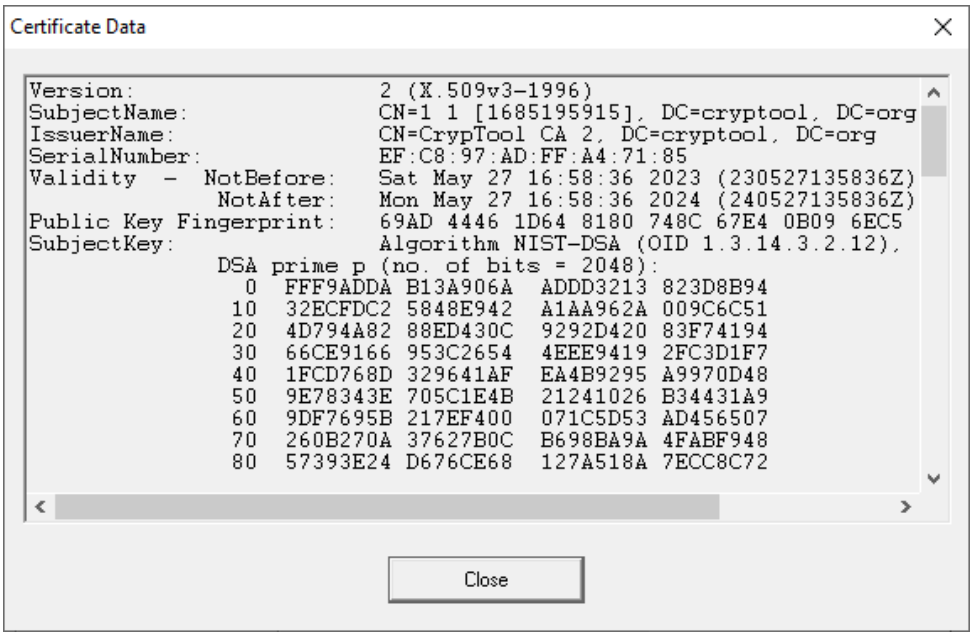


Рисунок 2 – Открытый ключ DSA-2048

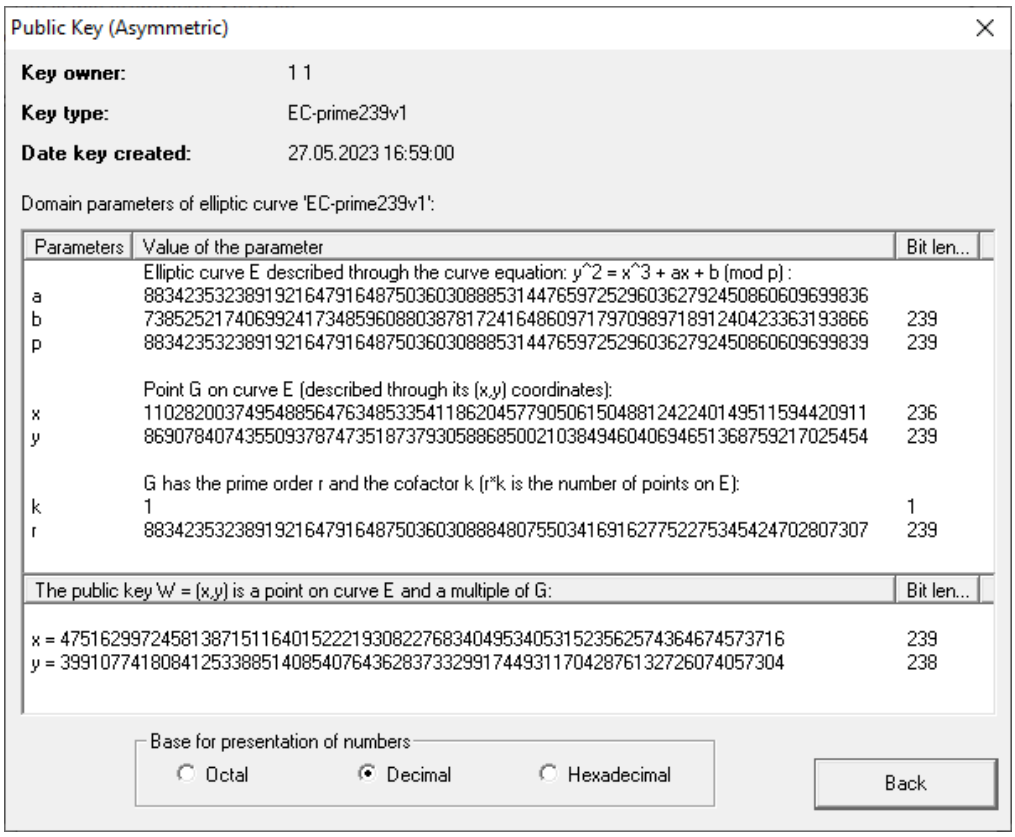


Рисунок 3 – Открытый ключ EC-239

2. Процессы создания и проверки электронной подписи

2.1 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.

2.2 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.

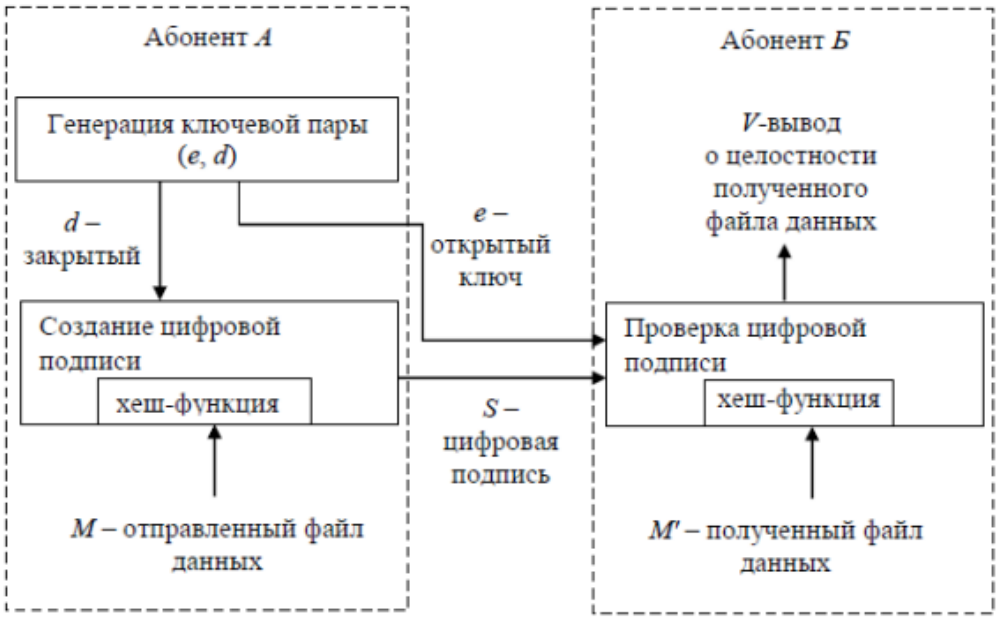


Рисунок 4 – Обобщенная схема создания и проверки электронной подписи

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.

2.3 Таблица с фактическими временами создания электронной подписи различными алгоритмами

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

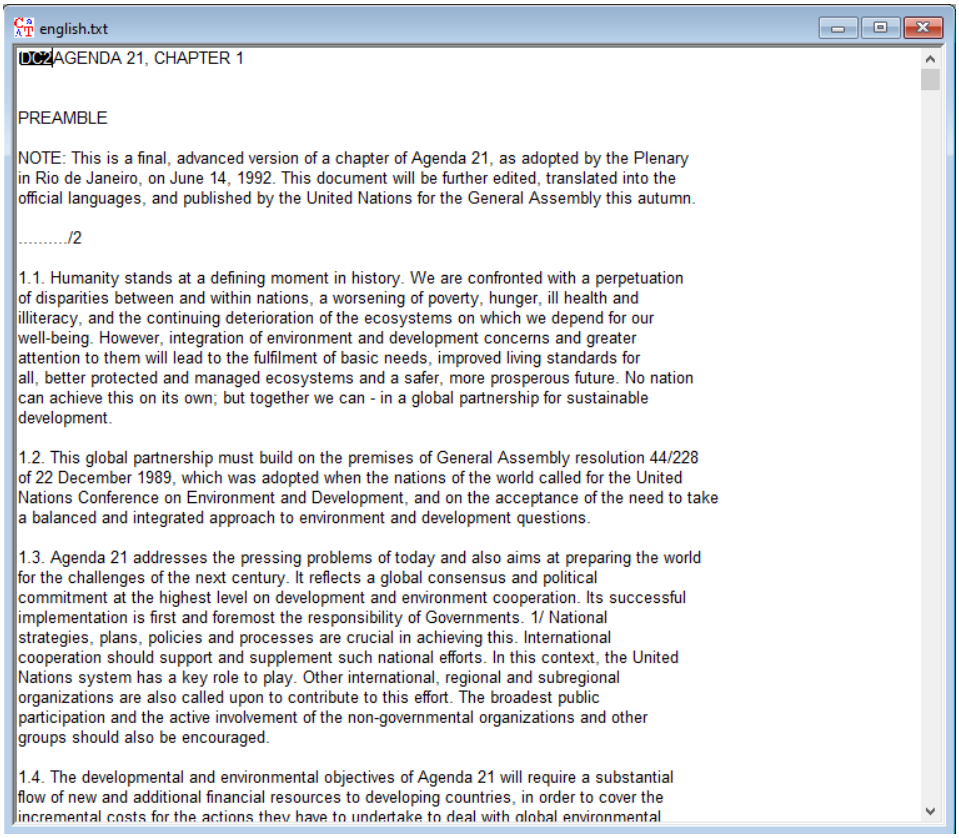


Рисунок 5 – Исходный текст

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

2.4 Скриншот со значением электронной подписи

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

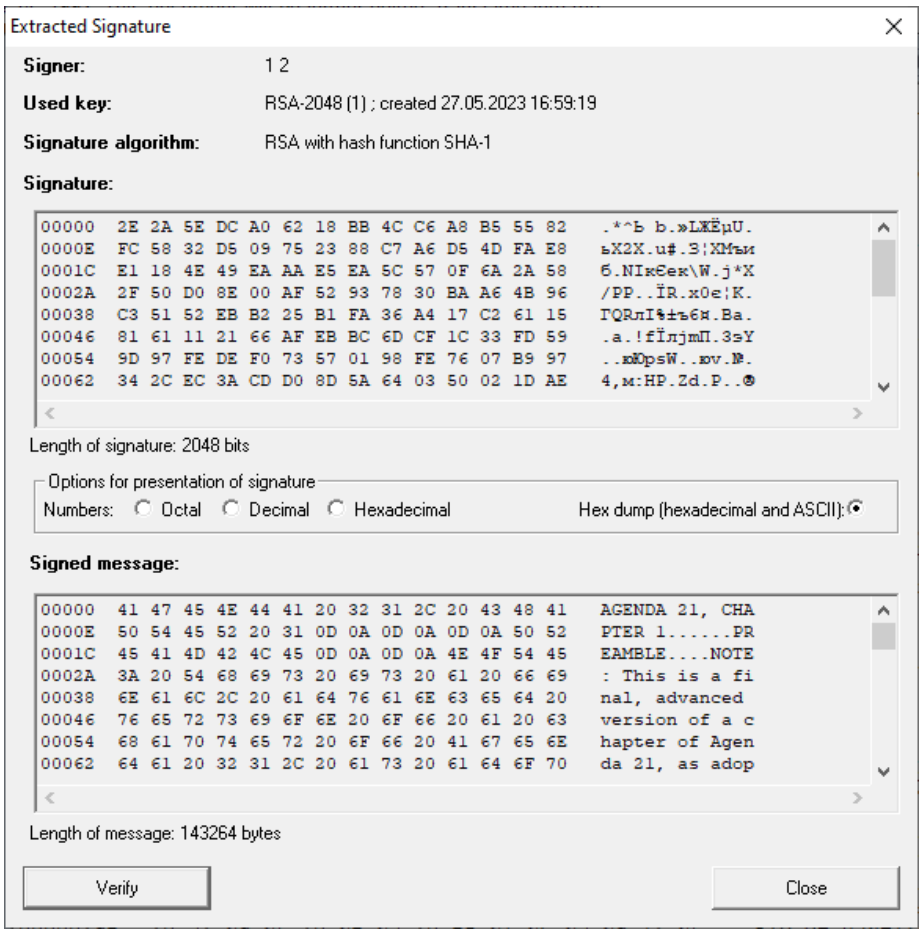


Рисунок 6 – Цифровая подпись

2.5 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

3. Создание и проверка электронной подписи на основе эллиптических кривых

3.1 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.

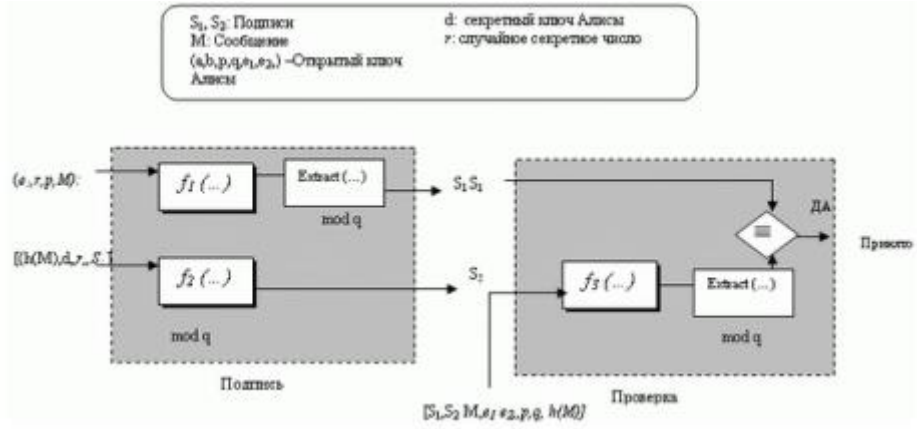


Рисунок 9 – Схема цифровой подписи ECDSA

Функция f_1 создает новую точку для секретного ключа подписывающего лица. Функция f_2 создает новую точку из двух общедоступных ключей подписывающего лица. Каждый экстрактор $Extract$ извлекает первые координаты соответствующей точки в модульной арифметике.

В процессе подписания две функции f_1 и f_2 и экстрактор (извлекающее устройство) создают две части подписи. В процессе проверки (верификации) обрабатывают выход одной функции f_2 (после прохождения через экстрактор) и сравнивают ее с первой частью подписи.

С использование ключевой пары (закрытый ключ – d, и открытый ключ – (a, b, p, q, e_1, e_2)) 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, $1 < r < q - 1$;

2. Выбирается третья точка на кривой $P(u, v) = r \times e_1$;

3. Используем абсциссу uu, чтобы вычислить первую часть подписи $S_1 = u \bmod q$;

4. Используем дайджест сообщения $h(M)$, закрытый ключ d, секретное случайное число r и S_1, чтобы вычислить вторую часть подписи $S_2 = (h(M) + d \times S_1) \times r - 1 \bmod q$.

Алгоритм проверки цифровой подписи ECDSA включает следующие операции:

1. Используем M, S_1, S_2 для получения промежуточных результатов A и B: $A = h(M) \times S_2^{-1} \bmod q, B = S_2 - 1 \times S_1 \bmod q$;

2. Затем восстанавливаем третью точку $T(x, y) = A \times e_1 + B \times e_2$;

3. Верификатор $V = x \bmod q$ сравниваем с S_1.

Схема протокола ECDSA представлена на рисунке 10.

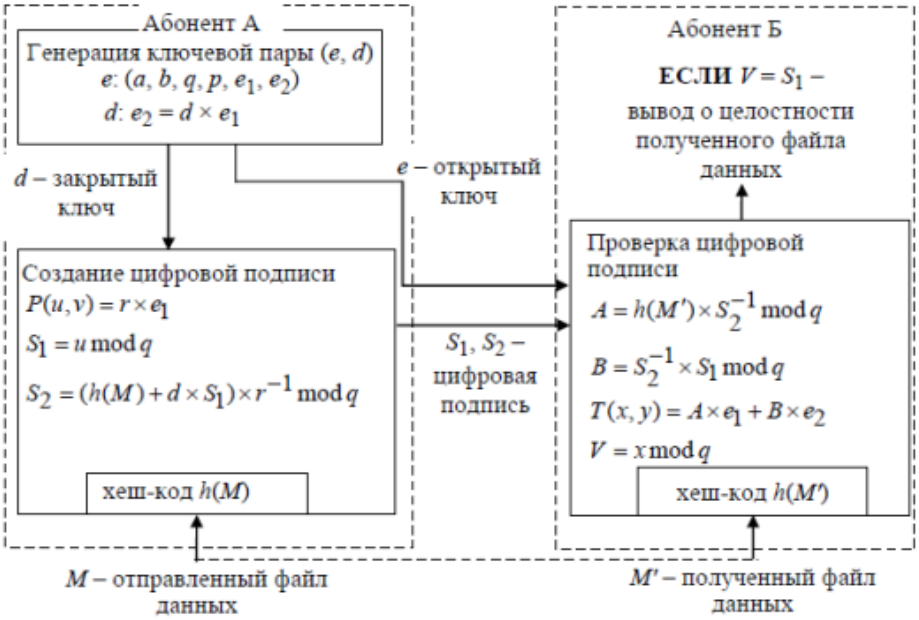


Рисунок 10 – Схема протокола ECDSA

3.2 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.

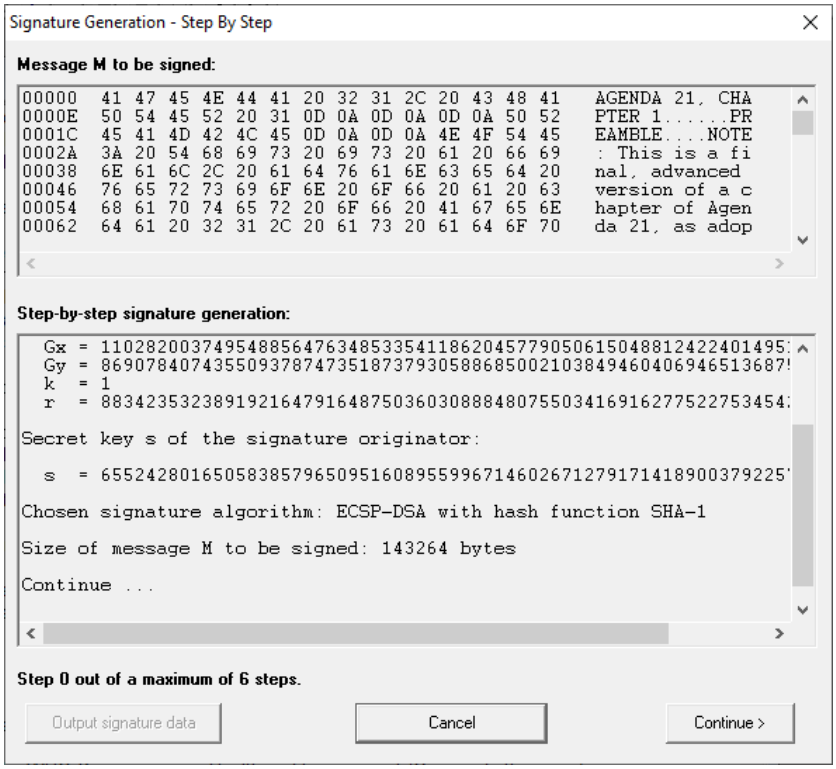


Рисунок 11 – Нулевой шаг создания подписи

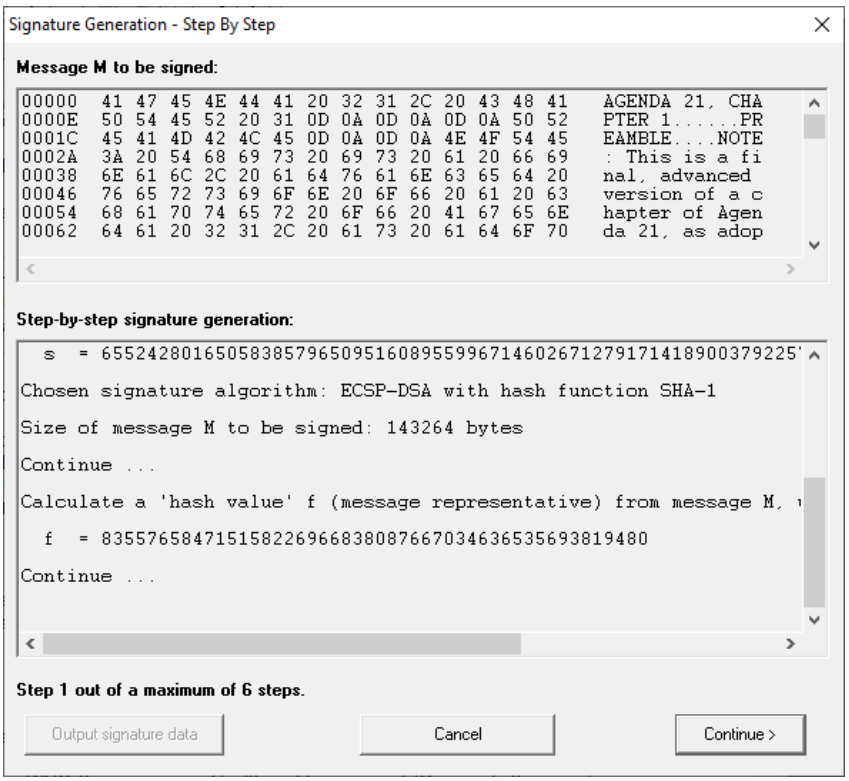


Рисунок 12 – Первый шаг создания подписи

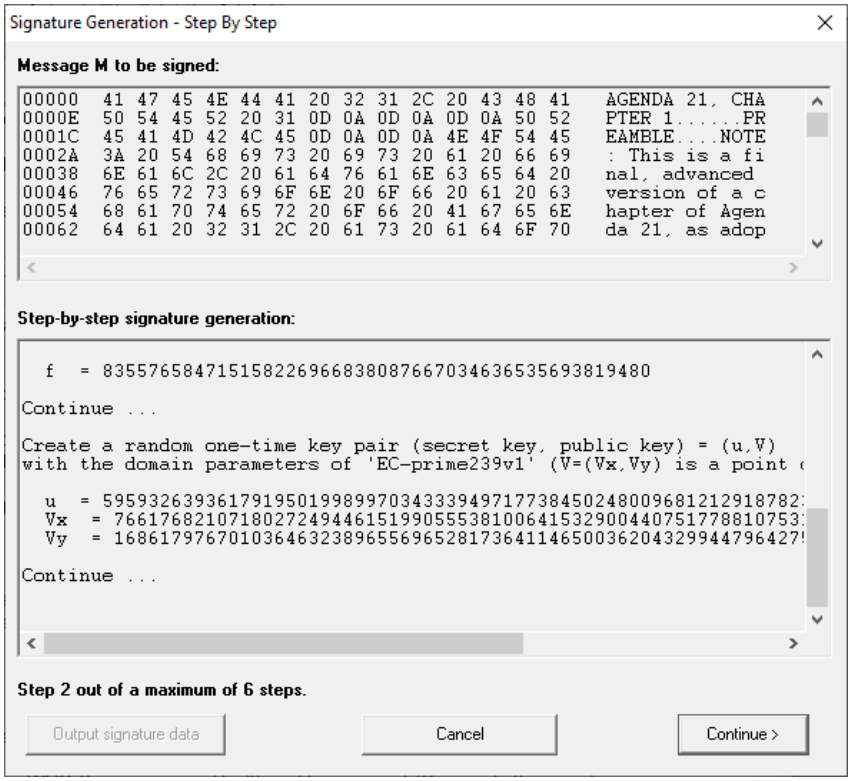


Рисунок 13 – Второй шаг создания подписи

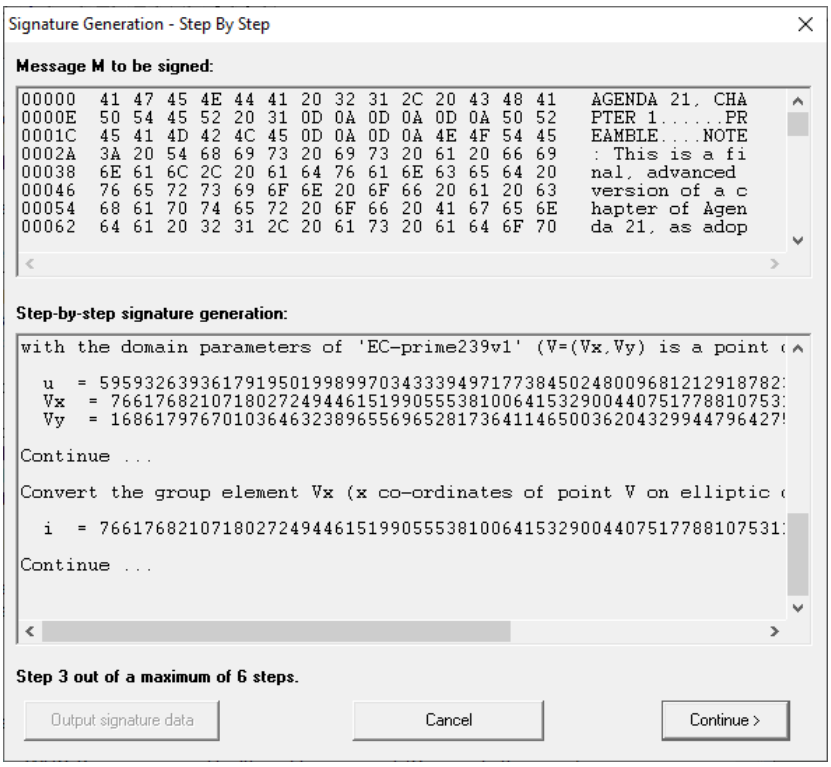


Рисунок 14 – Третий шаг создания подписи

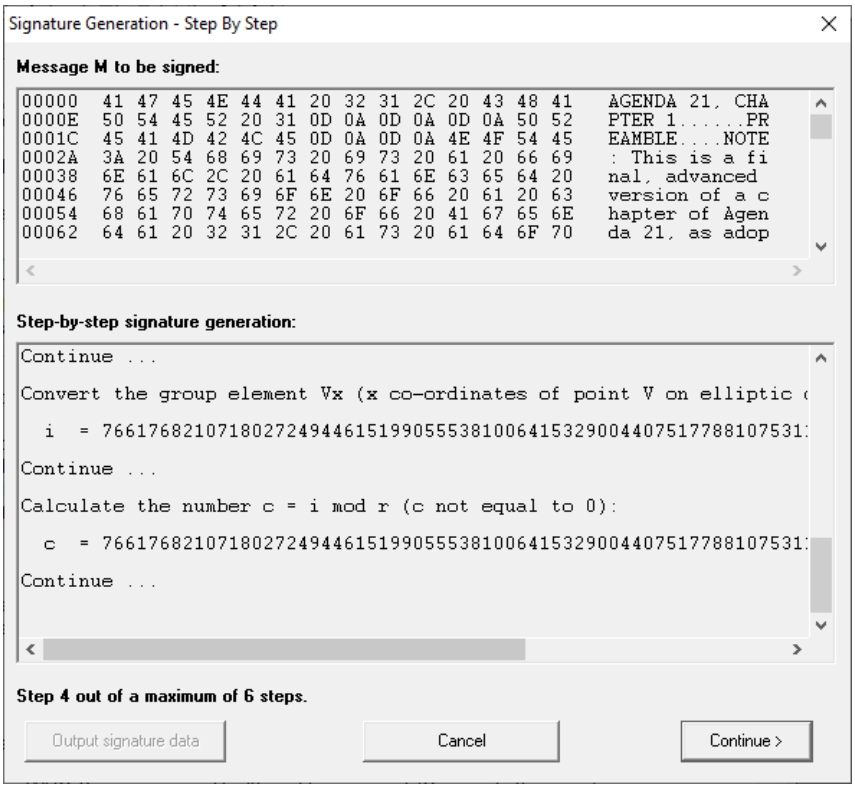


Рисунок 15 – Четвертый шаг создания подписи

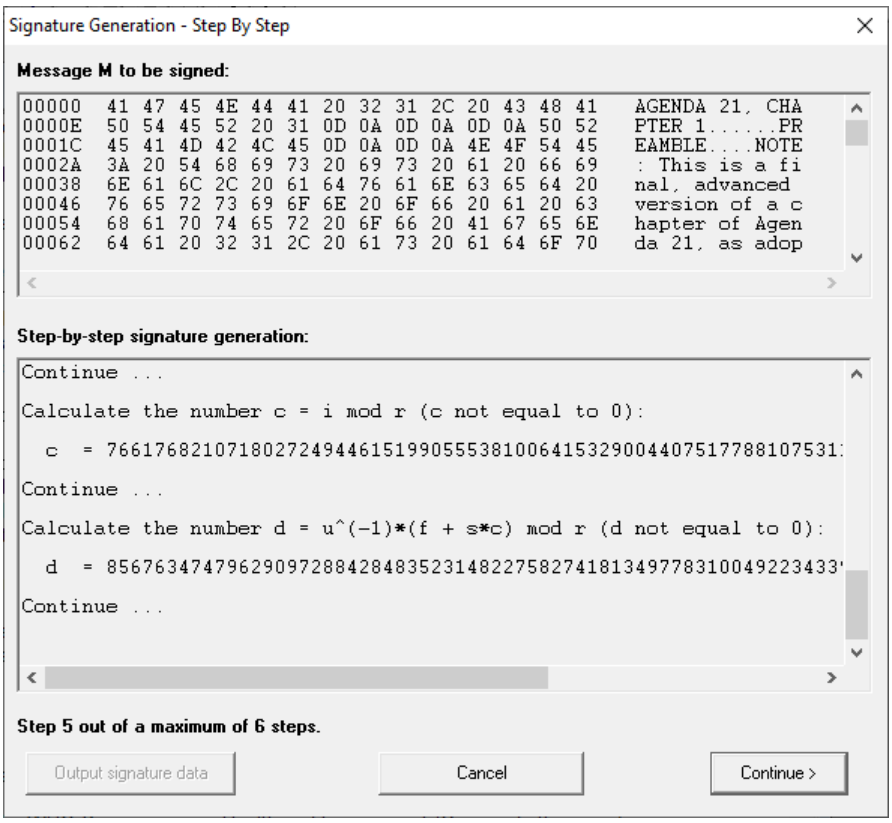


Рисунок 16 – Пятый шаг создания подписи

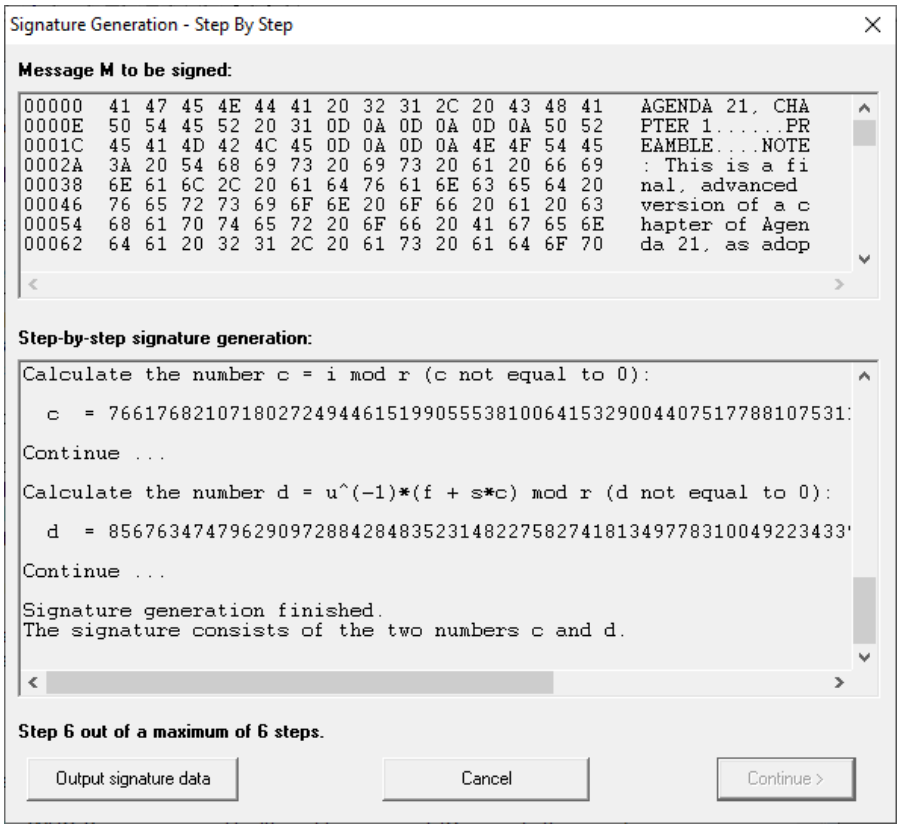


Рисунок 17 – Шестой шаг создания подписи

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
a	a
b	b
$e_1 = (x, y)$	(Gx, Gy)
q	r
d	s
$h(M)$	f
r	u
u	i
$P(u, v)$	(Vx, Vy)
S_1	c
S_2	d

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: $E_{23}(-18,75)$. Далее выберем точку $e_1 = (x_1, y_1) = (7, 4)$, как представлено на рисунке 20.

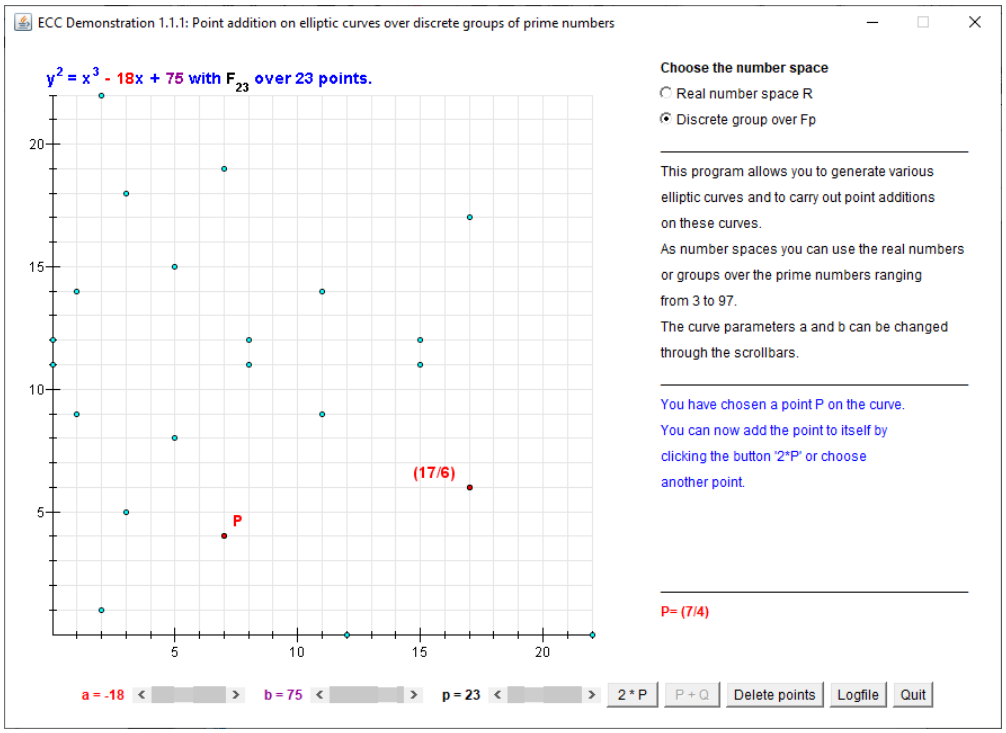


Рисунок 20 – Выбор точки e_1

Теперь подберем такое число q, что $q \times (x_1, y_1) = O$. Получилось, что $q = 11$, как продемонстрировано на рисунке 21.

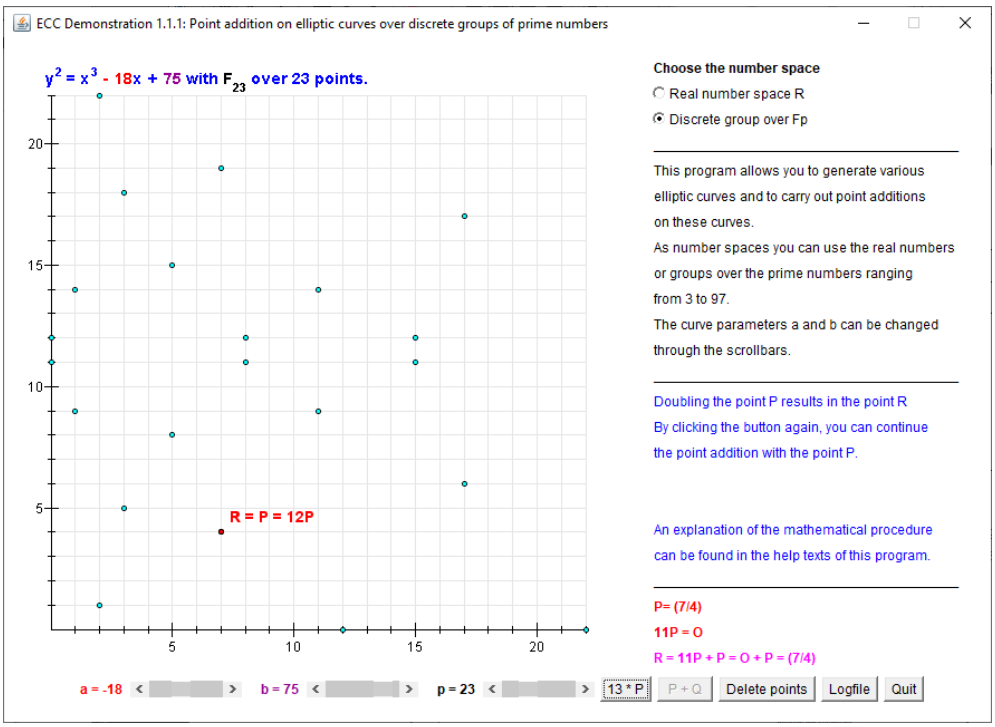


Рисунок 21 – Подбор числа q

Выбираем целое число dd такое, что $1 < d < q - 1$. Выбрано $d = 5$ – закрытый ключ. После чего, вычисляем точку $e_2 = d \times e_1 = 5 \times (7,4) = (1,9)$, как показано на рисунке 22.

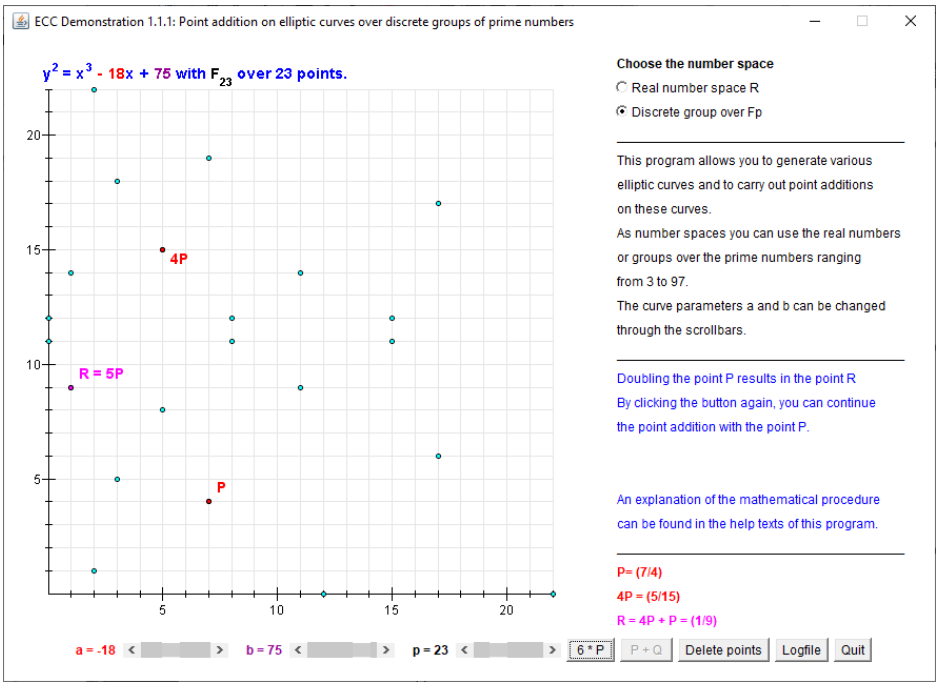


Рисунок 22 – Вычисление точки e_2

Открытый ключ – $(a = -18, b = 75, p = 23, q = 11, e_1 = (7,4), e_2 = (1,9))$.

2. ECDSA подписание.

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

Находим третью точку $P = r \times e_1 = 3 \times (7,4) = (15,11) \Rightarrow u = 15, v = 11$.

Вычисляем первую часть подписи: $S_1 = u \; mod \; q = 15 \; mod \; 11 = 4$.

Вычисляем вторую часть подписи: $S_2 = (h(M) + d \times S_1) \times r - 1 \; mod \; q = (66 + 5 \times 4) \times 3 - 1 mod \; 11 = 3$.

Тогда получаем $M = h(M) = 66, SS1 = 4, SS2 = 3$.

3. ECDSA проверка.

Вычислим промежуточные результаты A и B:

$A = h(M) \times S_2^{-1} mod \; q = 66 \times 3 - 1 \; mod \; 11 = 0$;

$B = S_2^{-1} \times S_1 \; mod \; q = 3^{-1} \times 4 \; mod \; 11 = 5$.

Восстанавливаем третью точку $T(x,y)$: $T(x,y) = A \times e_1 + B \times e_2 = 0 \times (7,4) + 5 \times (1,9) = (15,11)$.

Вычисляем верификатор V: $V = x \; mod \; q = 15 \; mod \; 11 = 4$.

Так как $V = 4 = S_1$ следовательно, текст сохранил свою целостность

4. Демонстрация процесса подписи в среде PK

4.1 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.

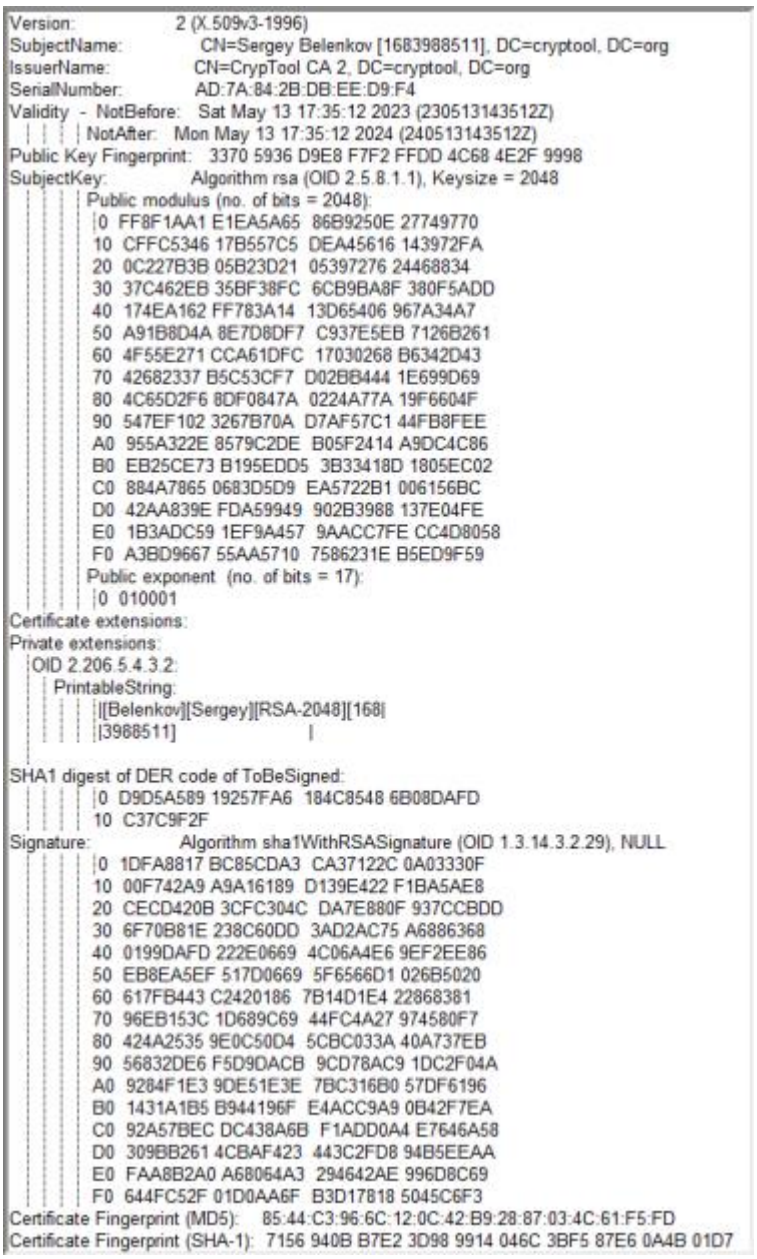


Рисунок 23 – Сертификат из CrypTool 1

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.

4.2 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.

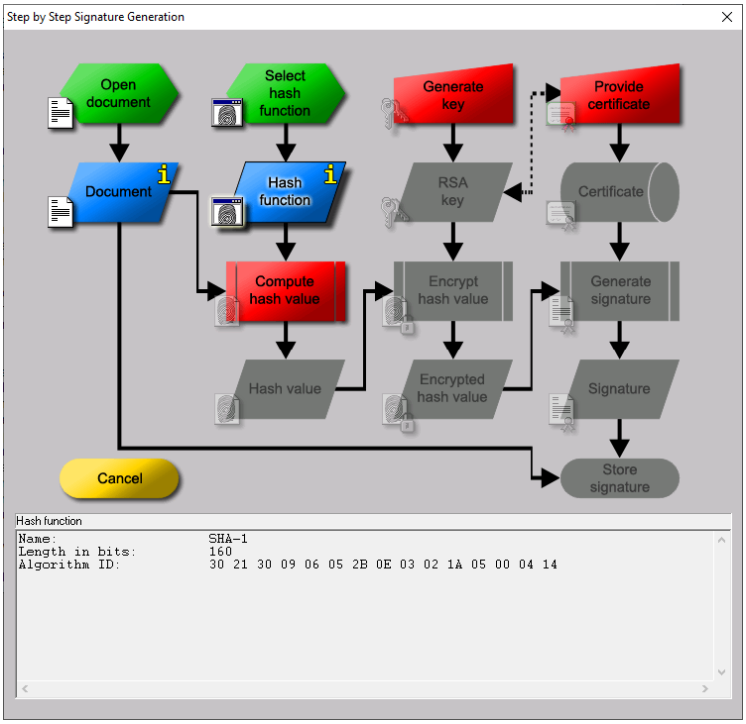


Рисунок 24 – Выбор хэш-функции

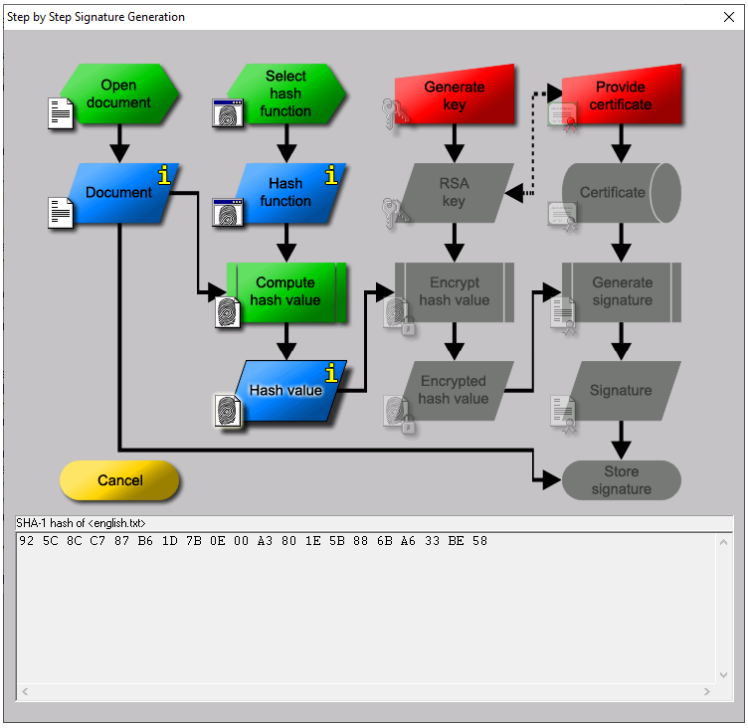


Рисунок 25 – Значение хэша документа

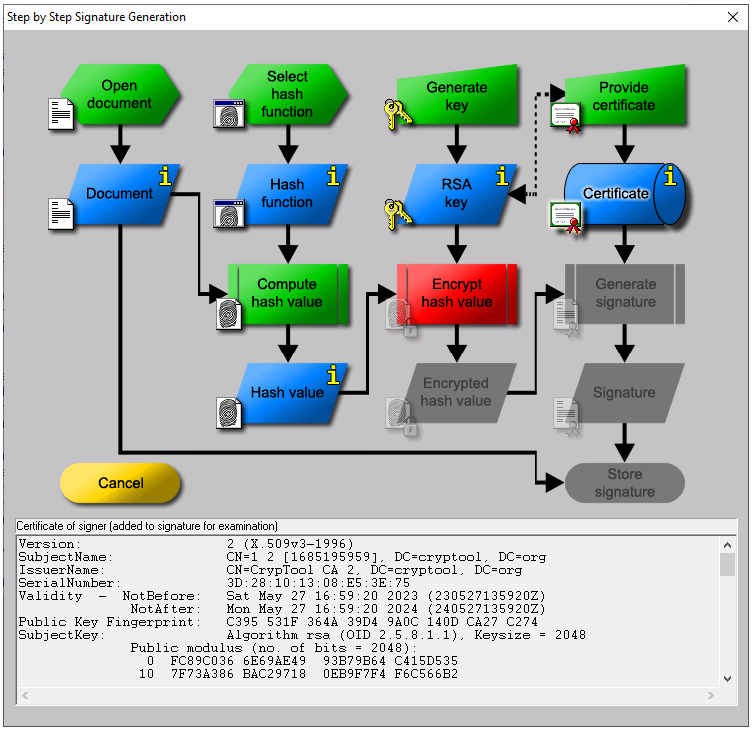


Рисунок 26 – Сертификат на сгенерированную ключевую пару

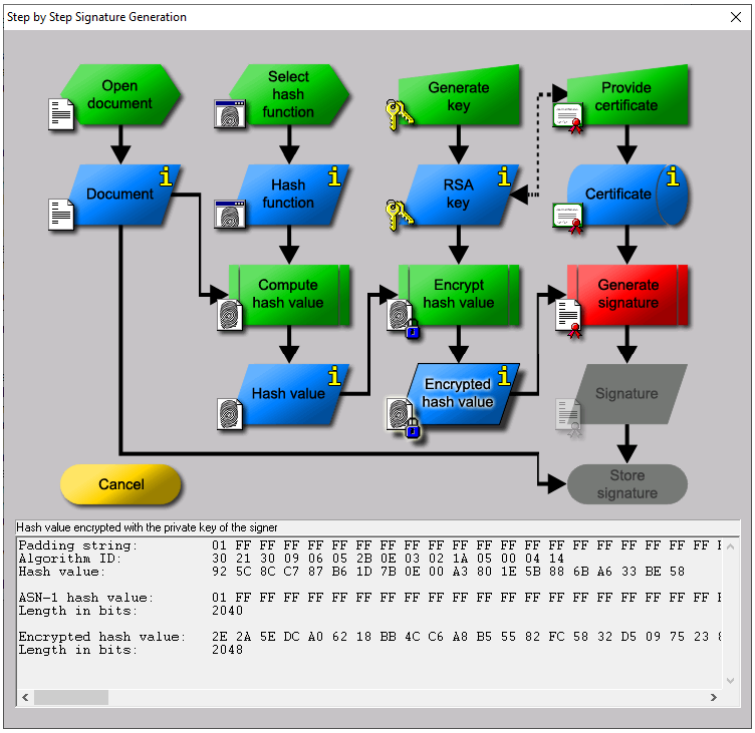


Рисунок 27 – Шифрование значения хэша документа

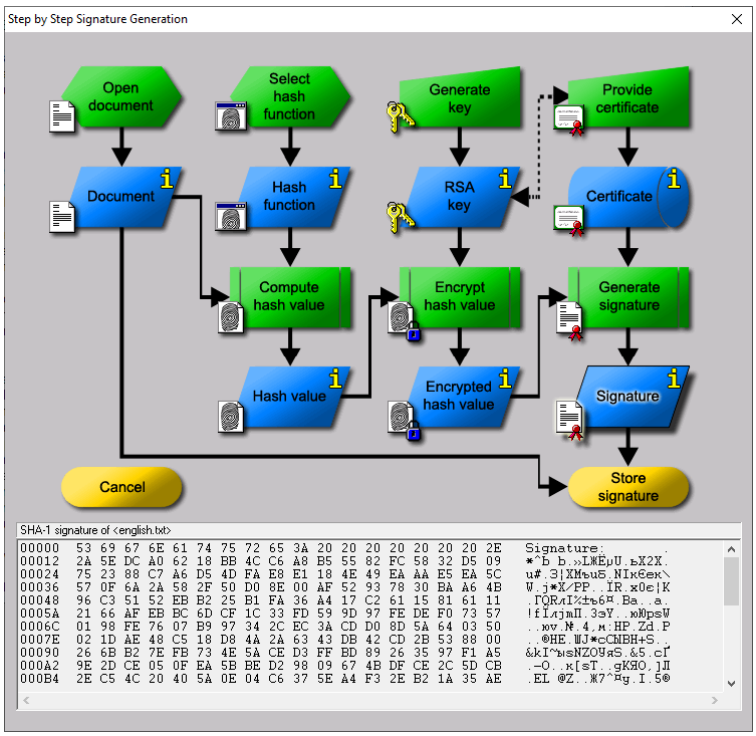


Рисунок 28 – Электронная подпись документа

5. Подписание своего отчета

5.1 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.

5.2 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.

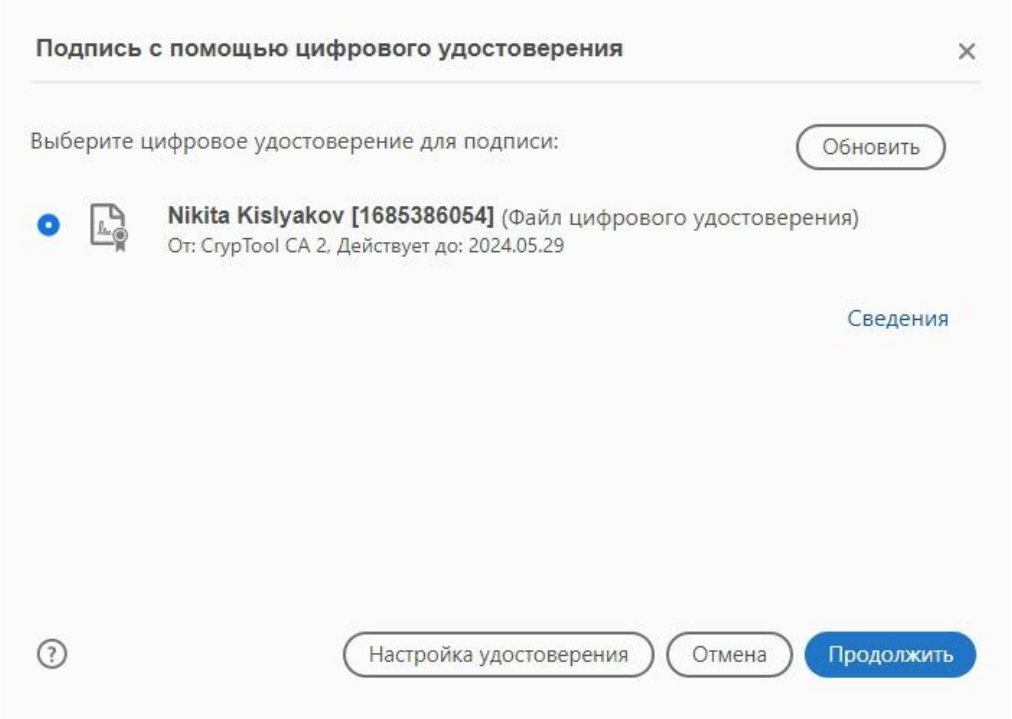


Рисунок 29 – Попытка подписания

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

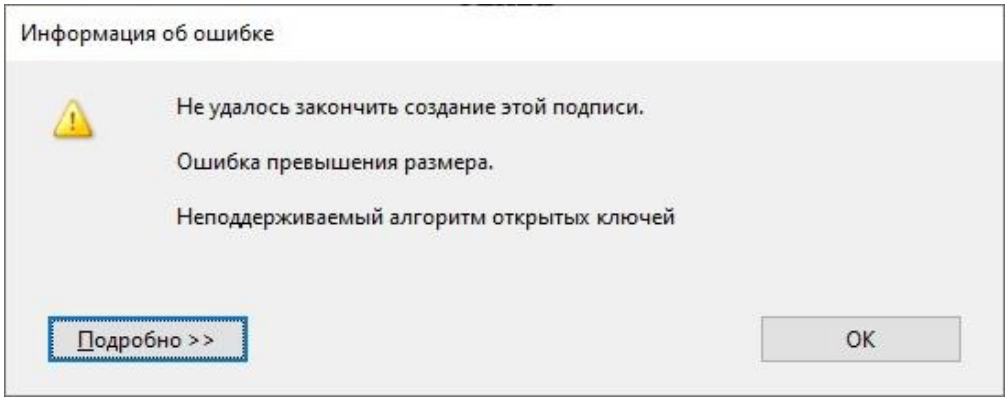


Рисунок 30 – Ошибка при попытке подписать файл

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

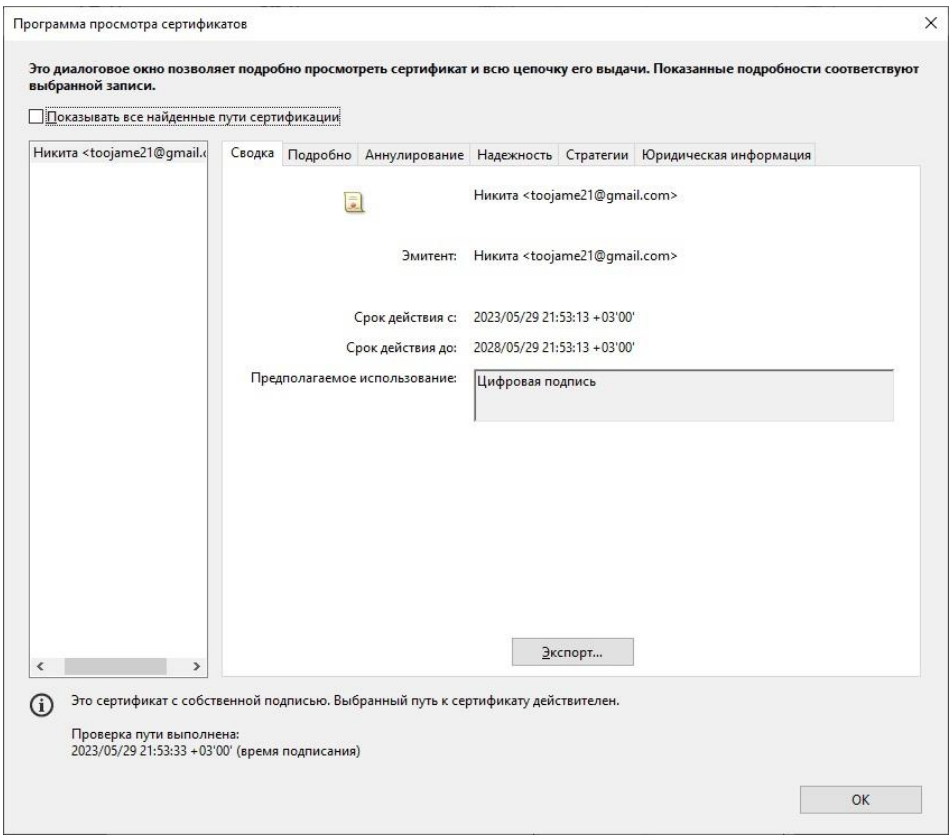
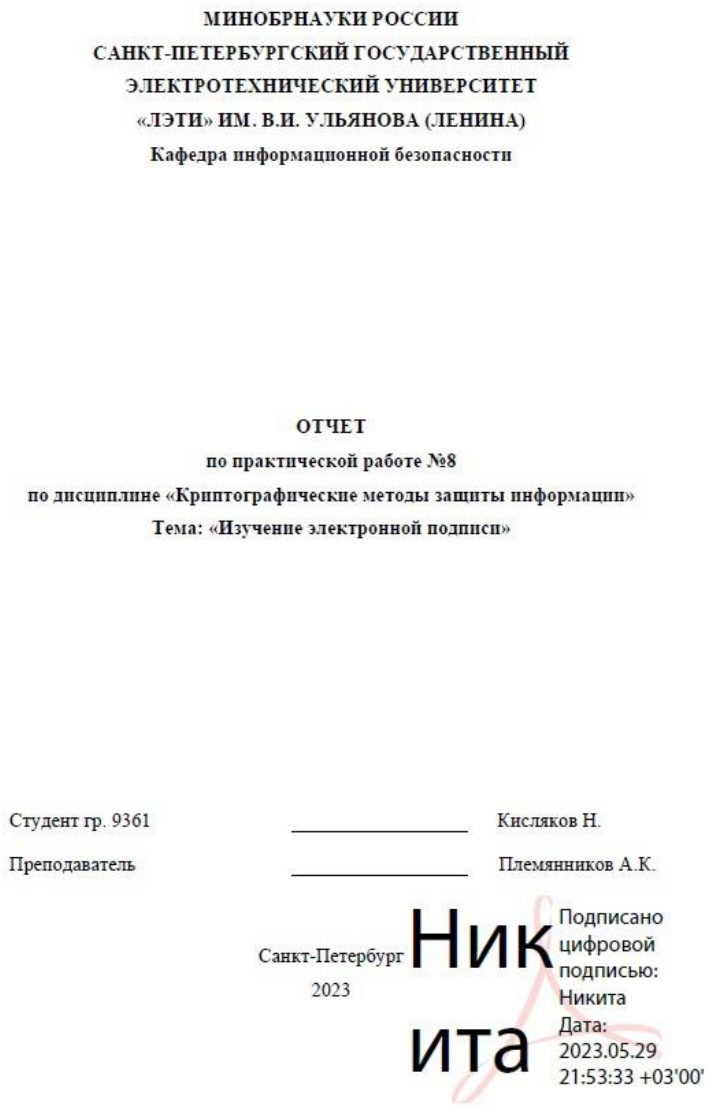


Рисунок 31 – Самоподписанный сертификат

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


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 А.К.

Санкт-Петербург
2023

Никита
Подписано цифровой подписью: Никита
Дата: 2023.05.29 21:53:33 +03'00'

Рисунок 32 – Титульный лист отчета с электронной подписью

5.3 Скриншоты свойств подписи и сертификата

Свойства подписи и сертификата продемонстрированы на рисунках 33 и 34.

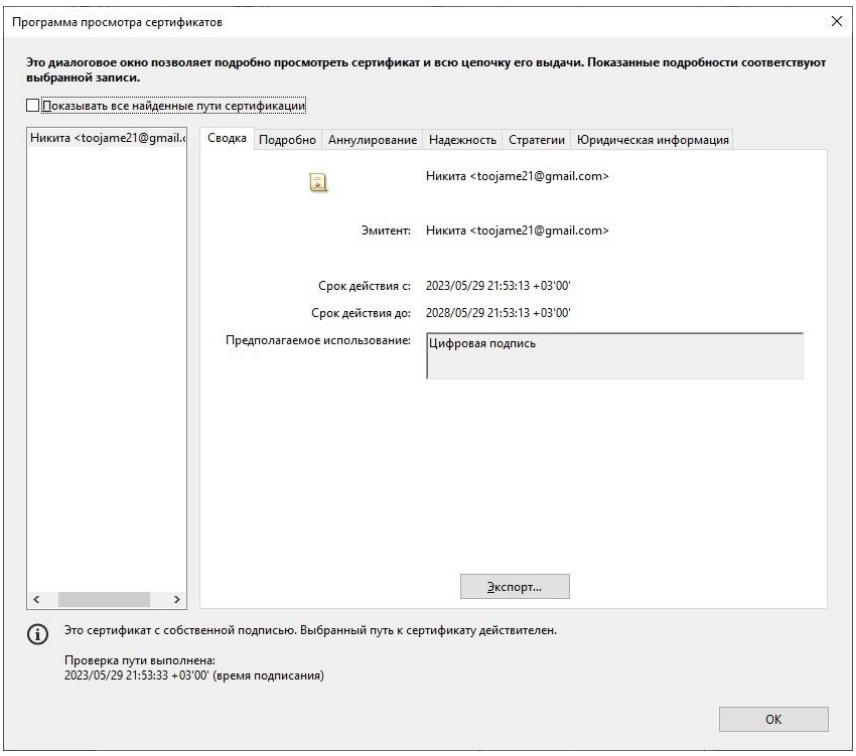


Рисунок 33 – Свойства сертификата

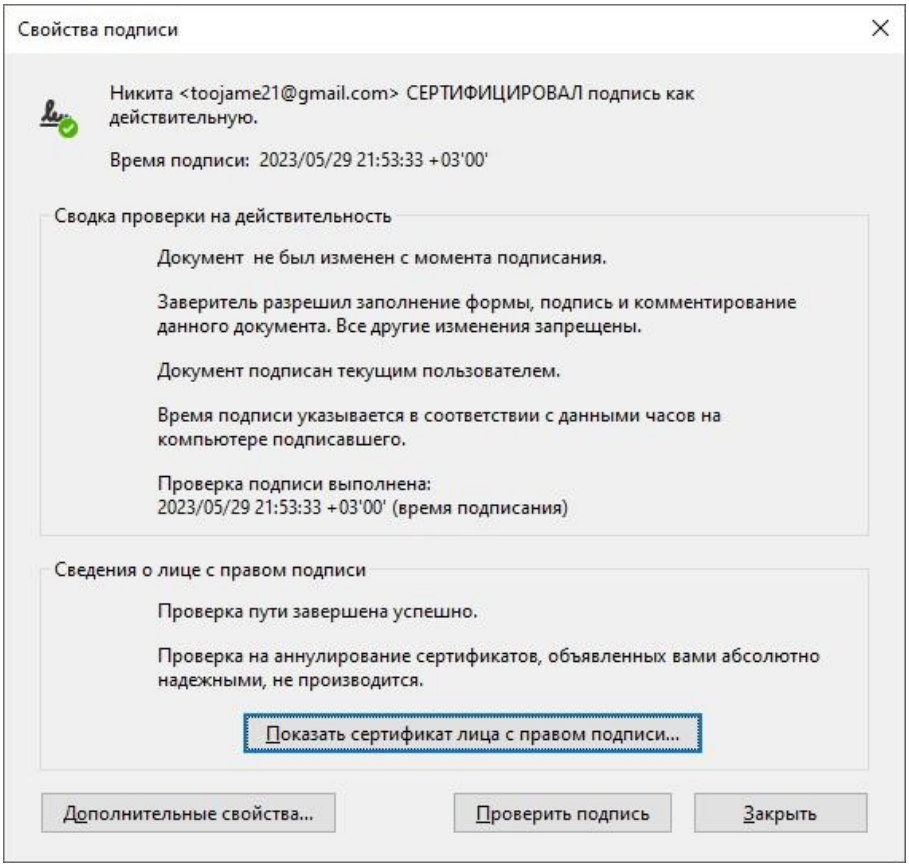


Рисунок 34 – Свойства подписи

5.4 Скриншот результата проверки после внесения изменений в отчет

Теперь внесем изменения в pdf-файл и проверим подпись файла. Результат проверки показан на рисунке 35.

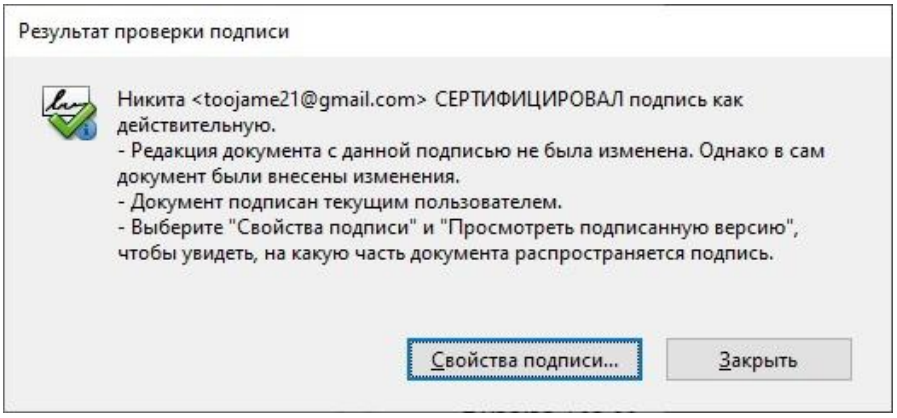


Рисунок 35 – Результат проверки электронной подписи

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 (e_1, e_2, p, q), а в качестве закрытого – значение d.

ECDSA: было определено, что в качестве открытого ключа алгоритм генерирует набор значений (a, b, p, q, e_1, e_2), 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), а RSA-2048 – наибольшее (1,721 секунд). Алгоритм DSA-2048 показал время 1,207 секунды

2. 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.

3. 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 (M, S_1, S_2) при помощи известного открытого ключа – (a, b, p, q, e_1, e_2) и закрытого ключа – d. M – это подписанные данные, а S_1, S_2 – части подписи. Для вычисления S_1 и S_2 выбирается секретное случайное число r, лежащее в диапазоне $1 < r < q - 1$. Далее находится точка на эллиптической кривой $P(u, v) = r \cdot e_1$. Значение S_1 вычисляется как $S_1 = u \; mod \; q$, а значение S_2 вычисляется как $S_2 = (h(M) + d \times S_1) \times r - 1 \; mod \; q$, 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 S_1. Если их значения совпадают, значит данные не были изменены. Набор значений (M, S_1, S_2) получается из электронной подписи и используется для получения промежуточных результатов A и B: $A = h(M) \times S_2 - 1 \; mod \; q, B = S_2 - 1 \times S_1 \; mod \; q$. Затем восстанавливается третья точка $T(x, y) = A \times e_1 + B \times e_2$. Далее вычисляется значение верификатора $V = \; x \; mod \; q$

4. 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.

5. 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.