МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №8
по дисциплине «Криптография и защита информации»
Тема: Изучение цифровой подписи

Студент гр. 9381 _____ Колованов Р.А.

Преподаватель _____ Племянников А.К.

Санкт-Петербург
2022

Цель работы.

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

Основные теоретические положения.

Генератор ключевых пар.

Генерация ключевых пар для алгоритма RSA.

1. Генерация двух больших простых чисел p и q;
2. Вычисление $n = p * q$;
3. Выбор произвольного e ($e < n$), взаимно простого с $\varphi(n)$;
4. Вычисление d: $e * d = 1 \ mod \ \varphi(n)$;
5. Числа (e, n) – открытый ключ, d – закрытый ключ, p и q уничтожаются.

Генерация ключевых пар для алгоритма DSA.

1. Выбирается число p: длина – [512, 1024] битов и число битов в p должно быть кратно 64;
2. Выбирается число q, которое имеет тот же самый размер дайджеста 160 битов, такое, что: $(p - 1) = 0 \ mod \ q$;
3. Выбирается e_1: $e_1^q = 1 \ mod \ p$;
4. Выбирается целое число $d < q$ и вычисляется $e_2 = e_1^d \ mod \ p$;
5. Числа (e_1, e_2, p, q) – открытый ключ, d – закрытый ключ.

Генерация ключевых пар для алгоритма ECDSA.

1. Выбирается эллиптическая кривая $E_p(a, b)$, p − простое число;
2. Выбирается точка на кривой $e_1 = (x_1, y_1)$;
3. Выбирается простое число q – порядок одной из циклических подгрупп группы точек эллиптической кривой: $q \times (x_1, y_1) = O$;
4. Выбирается закрытый ключ d;

5. Вычисляется точка на кривой $e_2 = d \times e_1$;
6. Открытый ключ - (a, b, q, p, e_1, e_2).

Процессы создания и проверки цифровой подписи.

Обобщенные схемы подписания и проверки цифровой подписи представлены на рисунке 8.1.

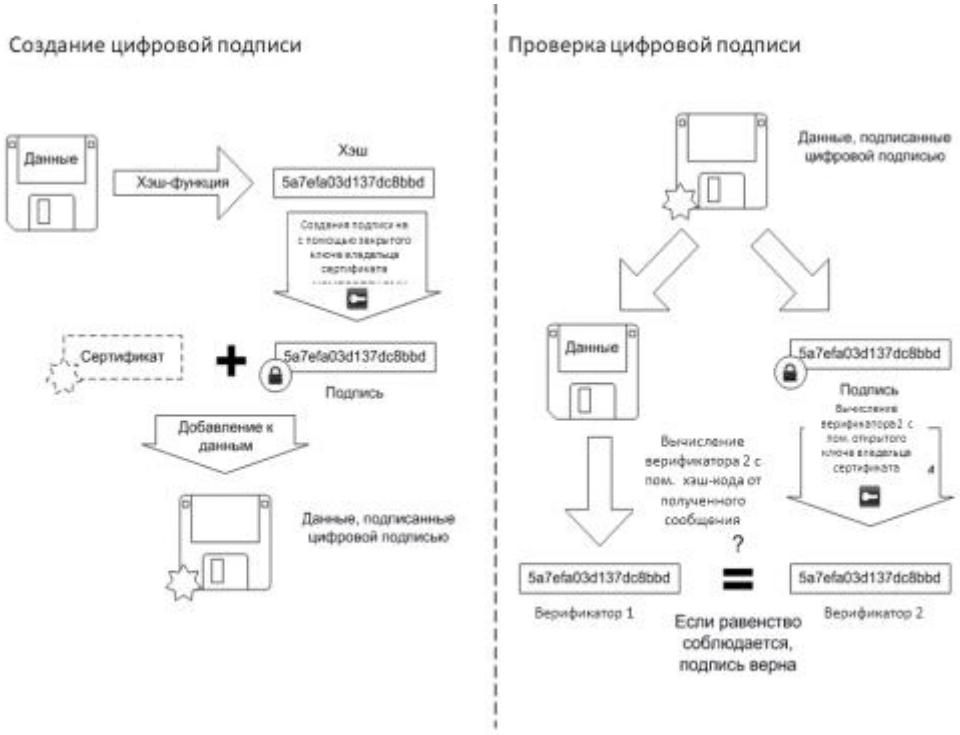


Рисунок 8.1

Схемы цифровой подписи на эллиптических кривых.

Схема цифровой подписи ECDSA (рисунок 8.2).

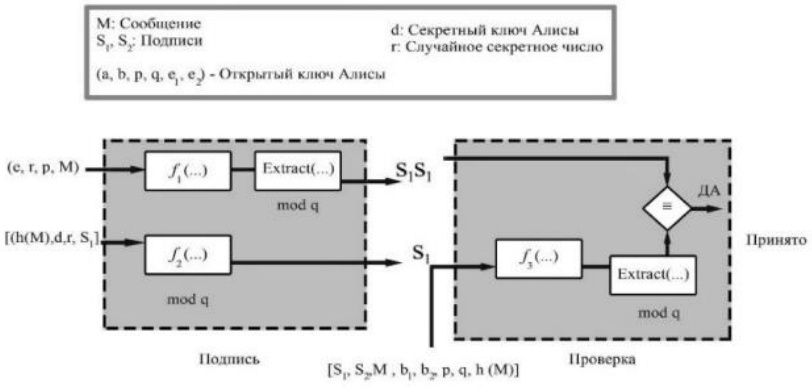


Рисунок 8.2

В процессе подписания две функции f_1 и f_2 и экстрактор *Extract* 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.

После того, как сгенерирована ключевая пара (закрытый ключ – d, и открытый ключ – (a, b, q, p, e_1, e_2), осуществляется подписание документа, затем на принимающей стороне осуществляется проверка (рисунок 8.3).

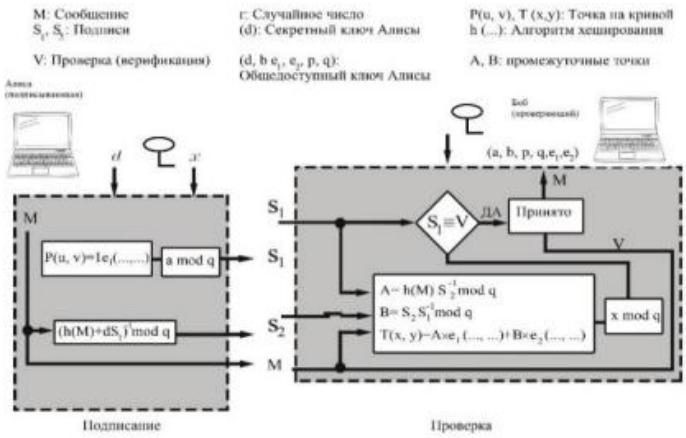


Рисунок 8.3

Алгоритм подписания ECDSA состоит из следующих операций:

1. Выбирается секретное случайное число r: $r \in (1, q-1)$;
2. Выбирается третья точка на кривой: $P(u, v) = r \times e_1$;
3. Вычисляется первая часть подписи по формуле:

$$S_1 = u \bmod q ,$$

где u – абсцисса;

4. Вычисляется вторая часть подписи по формуле:

$$S_2 = (h(M) + d \times S_1) \times r^{-1} \bmod q ,$$

где $h(M)$ – дайджест сообщения, d – закрытый ключ.

Алгоритм проверки цифровой подписи ECDSA включает следующие операции:

1. Вычисляем промежуточные результаты A и B:

$$A = h(M) \times {S_2}^{-1} mod\ q$$

$$B = {S_2}^{-1} \times S_1\ mod\ q$$

2. Восстанавливаем третью точку:

$$T(x, y) = A \times e_1 + B \times e_2$$

3. Верификатор $V = x\ mod\ q$ сравнивается с первой частью цифровой подписи S_1.

Демонстрация процесса подписи в среде PKI.

Инфраструктура открытых ключей (ИОК, PKI – Public Key Infrastructure) – набор средств (технических, материальных, организационных и т. д.), распределённых служб и компонентов, в совокупности используемых для поддержки решения основных задач криптографии, а именно:

1. Обеспечение конфиденциальности информации;
2. Обеспечение целостности информации;
3. Обеспечение аутентификации пользователей и ресурсов, к которым обращаются пользователи;
4. Обеспечение возможности подтверждения совершенных пользователями действий.

Решение перечисленных задач основано на использовании сертификатов открытых ключей. Сертификат открытого ключа – это электронный документ, который содержит:

1. Открытый ключ пользователя;
2. Информацию о пользователе, которому принадлежит сертификат;
3. Информацию о сроке действия сертификата;
4. Информацию об издателе сертификата;
5. Другие атрибуты;
6. Цифровую подпись этих данных, созданную удостоверяющим центром, издавшим и выдавшим этот сертификат.

Существует несколько вариантов использования сертификатов открытых ключей:

1. Для зашифрования и расшифрования электронных документов;
2. Для подписания электронного документа и проверки подписи;
3. Для аутентификации отправителя документа.

Ход работы.

Генераторов ключевых пар.

Задание.

1. Перейти к утилите «Digital Signatures / PKI -> PKI -> Generate/Import Keys»;
2. Сгенерировать ключевые пары по алгоритмам RSA-2048, DSA-2048, EC-239. Зафиксируйте время генерации в таблице;
3. С помощью утилиты «Digital Signatures / PKI -> PKI -> Display/Export Keys» вывести сгенерированный открытый ключ и сохранить соответствующий скриншот.

Описание алгоритмов генерации.

Генерация ключей RSA-2048:

1. Генерация двух больших простых чисел p и q (p и q держаться в секрете);
2. Вычисление $n = p * q$;
3. Выбор произвольного e ($e < n$), взаимно простого с $\varphi(n) = (p - 1)(q - 1)$ – функцией Эйлера;
4. Вычисление закрытого ключа d: $e * d = 1 \ mod \ \varphi(n)$;
5. Числа (e, n) – открытый ключ, d – закрытый ключ, p и q уничтожаются.

Генерация ключей DSA-2048:

1. Выбирается простое число p, 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 ${e_2} = {e_1}^d \ mod \ p$;
5. Числа (e_1, e_2, p, q) – открытый ключ, d – закрытый ключ.

Генерация ключей ECDSA (EC-239):

1. Выбирается эллиптическая кривая $E_p(a, b)$, p − простое число;
2. Выбирается точка на кривой $e_1 = (x_1, y_1)$;
3. Выбирается простое число q – порядок одной из циклических подгрупп группы точек эллиптической кривой: $q \times (x_1, y_1) = O$;
4. Выбирается целое число d ($1 < d <$ q – 1), и назначается закрытым ключом;
5. Вычисляется точка на кривой $e_2 = d \times e_1$;
6. Открытый ключ - (a, b, q, p, e_1, e_2).

Генерация ключевых пар.

Для начала при помощи утилиты «Digital Signatures / PKI -> PKI -> Generate/Import Keys» были сгенерированы ключевые пары по алгоритмам RSA-2048, DSA-2048 и EC-239, при этом так же было зафиксировано время генерации ключевых пар. Результаты представлены в таблице 1.

Таблица 1 – Время генерации ключевых пар.

Алгоритм	Время генерации пары ключей
RSA-2048	0.894 секунды
DSA-2048	2.899 секунды
EC-239 (prime239v1)	0.012 секунды

Как видно из результатов, алгоритм EC-239 показал наименьшее время генерации пары ключей, а DSA-2048 – наибольшее.

Сгенерированные открытые ключи представлены на рисунках 1, 2 и 3.

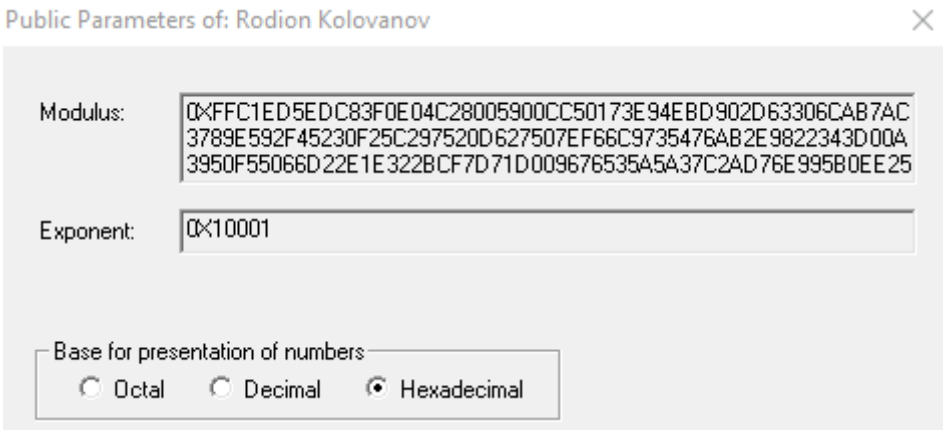


Рисунок 1 – Открытый ключ RSA-2048.

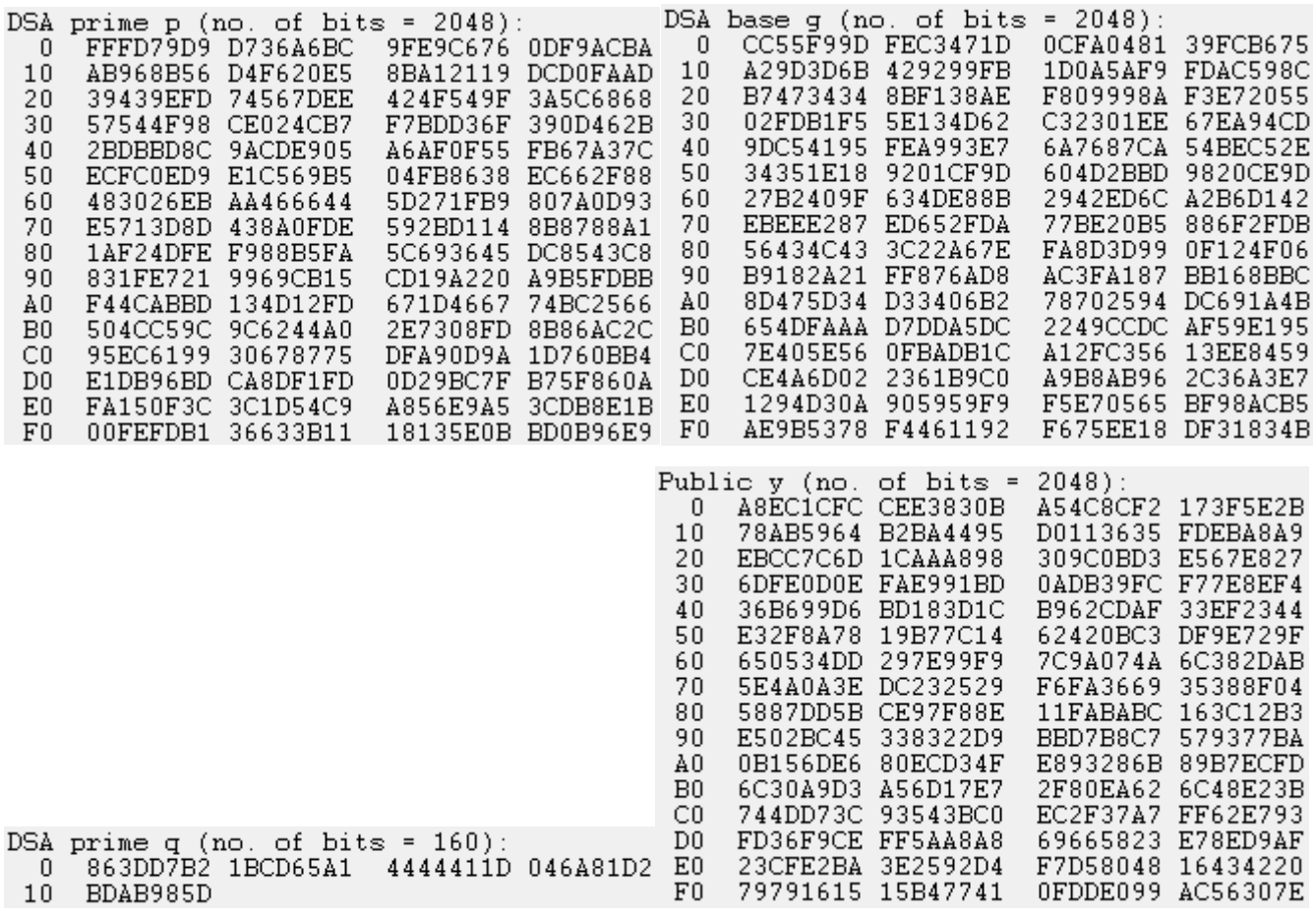
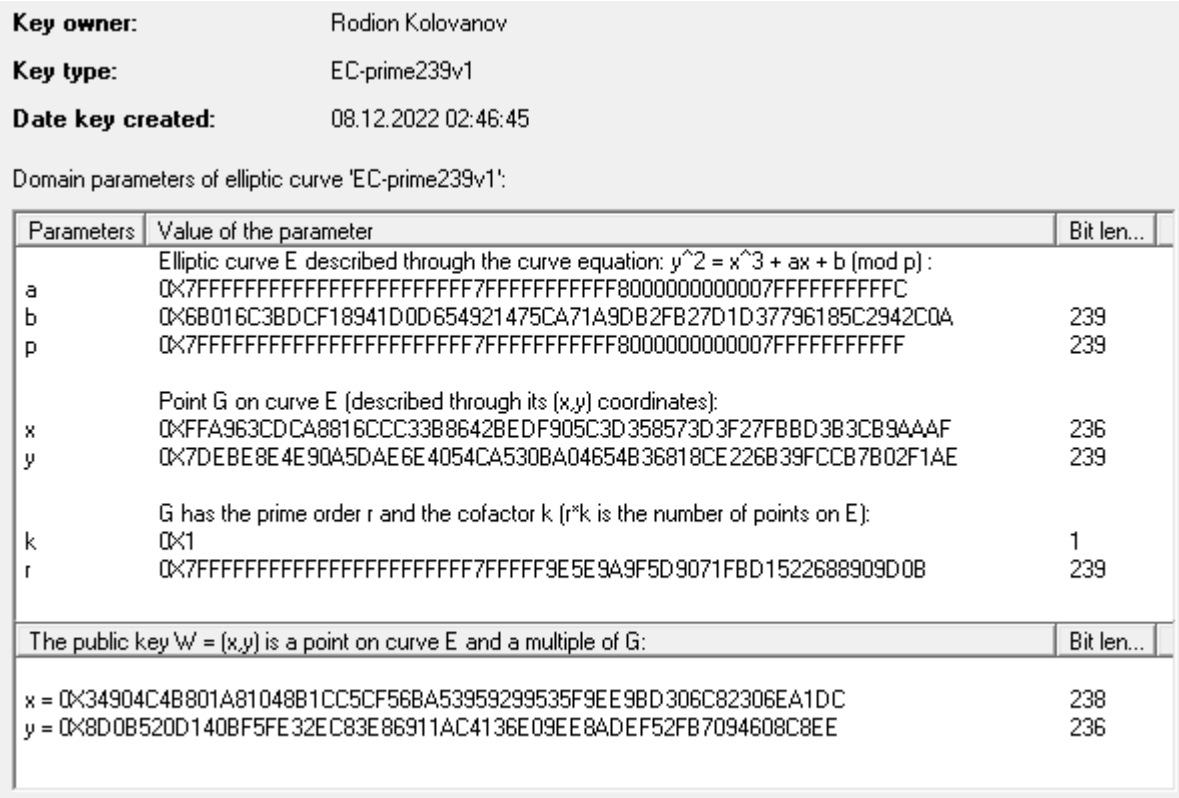


Рисунок 2 – Открытый ключ DSA-2048.



Key owner: Rodion Kolovanov

Key type: EC-prime239v1

Date key created: 08.12.2022 02:46:45

Domain parameters of elliptic curve 'EC-prime239v1':

Parameters	Value of the parameter	Bit len...
	Elliptic curve E described through the curve equation: y^2 = x^3 + ax + b (mod p) :	
a	0X7FFFFFFFFFFFFFFFFFFFFFFF7FFFFFFFFFFF8000000000007FFFFFFFFFFC	
b	0X6B016C3BDCF18941D0D654921475CA71A9DB2FB27D1D37796185C2942C0A	239
p	0X7FFFFFFFFFFFFFFFFFFFFFFF7FFFFFFFFFFF8000000000007FFFFFFFFFFF	239
	Point G on curve E (described through its (x,y) coordinates):	
x	0XFFA963CDCA8816CCC33B8642BEDF905C3D358573D3F27FBBD3B3CB9AAAF	236
y	0X7DEBE8E4E90A5DAE6E4054CA530BA04654B36818CE226B39FCCB7B02F1AE	239
	G has the prime order r and the cofactor k (r*k is the number of points on E):	
k	0X1	1
r	0X7FFFFFFFFFFFFFFFFFFFFFFF7FFFFF9E5E9A9F5D9071FBD1522688909D0B	239

The public key W = (x,y) is a point on curve E and a multiple of G:	Bit len...
x = 0X34904C4B801A81048B1CC5CF56BA53959299535F9EE9BD306C82306EA1DC	238
y = 0X8D0B520D140BF5FE32EC83E86911AC4136E09EE8ADEF52FB7094608C8EE	236

Рисунок 3 – Открытый ключ EC-239.

Процессы создания и проверки цифровой подписи.

Задание.

1. Открыть текст не менее 5000 знаков. Перейти к приложению «Digital Signatures/PKI -> Sign Document»;
2. Задайте хэш-функцию, и другие параметры цифровой подписи;
3. Создайте подпись ключами, сгенерированными в предыдущем задании. Зафиксируйте время создания цифровой подписи для каждого ключа;
4. Сохраните скриншот цифровой подписи с помощью приложения «Digital Signatures/PKI -> Extract Signature»;
5. Выполните процедуру проверки подписи «Digital Signatures/PKI -> Verify Signature» для случаев сохранения и нарушения целостности исходного текста. Сохраните скриншоты результатов.

Обобщенная схема создания и проверки цифровой подписи.

Обобщенная схема создания и проверки цифровой подписи представлена на рисунке 4.

Рисунок 4 – Схема создания и проверки цифровой подписи.

Создание и проверка цифровой подписи.

В качестве исходного текста был взят текст (размер текста превышает 5000 символов и составляет около 6300 символов), представленный на рисунке 5.

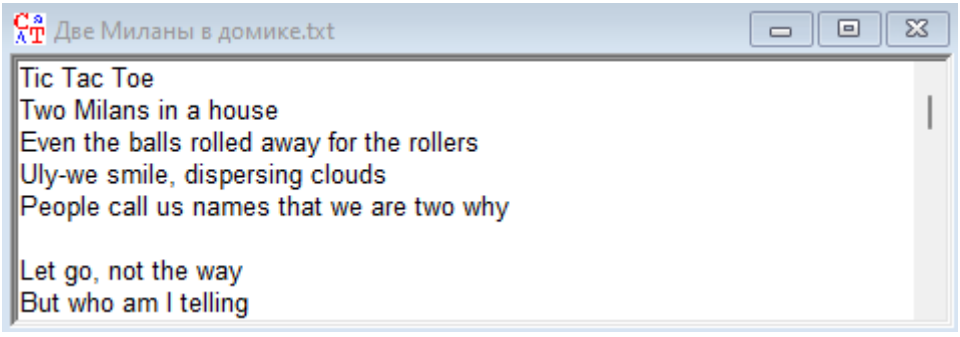


Рисунок 5 – Исходный текст.

Далее при помощи ранее сгенерированных ключевых пар были созданы цифровые подписи, при этом так же было зафиксировано время генерации цифровых подписей. Результаты представлены в таблице 2.

Таблица 2 – Время генерации цифровых подписей.

Алгоритм	Хэш-функция	Время генерации цифровой подписи
RSA-2048	SHA-1	0.010 секунд
DSA-2048	SHA-1	0.002 секунды
ECSP-DSA	SHA-1	0.002 секунды
ECSP-NR	SHA-1	0.002 секунды

Сгенерированные цифровые подписи представлены на рисунках 6, 7, 8 и 9.

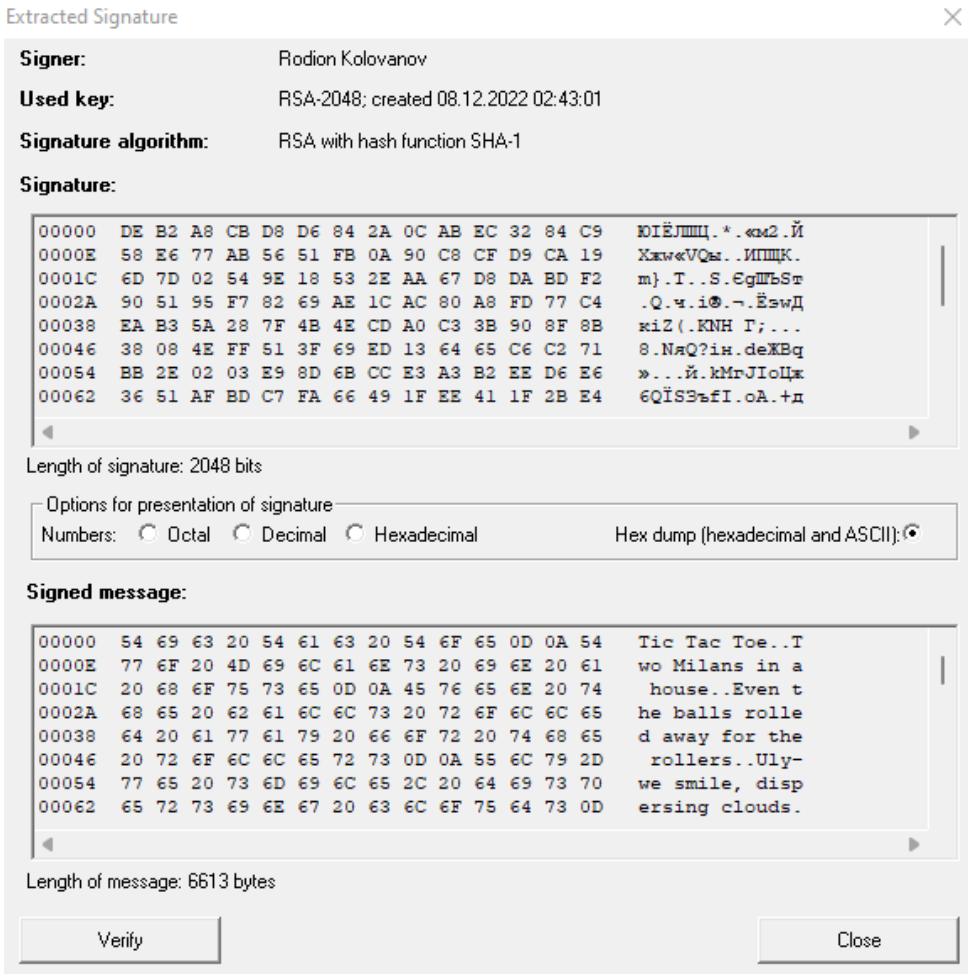


Рисунок 6 – Цифровая подпись, сгенерированная алгоритмом RSA с использованием хэш-функции SHA-1 и ключа RSA-2048.

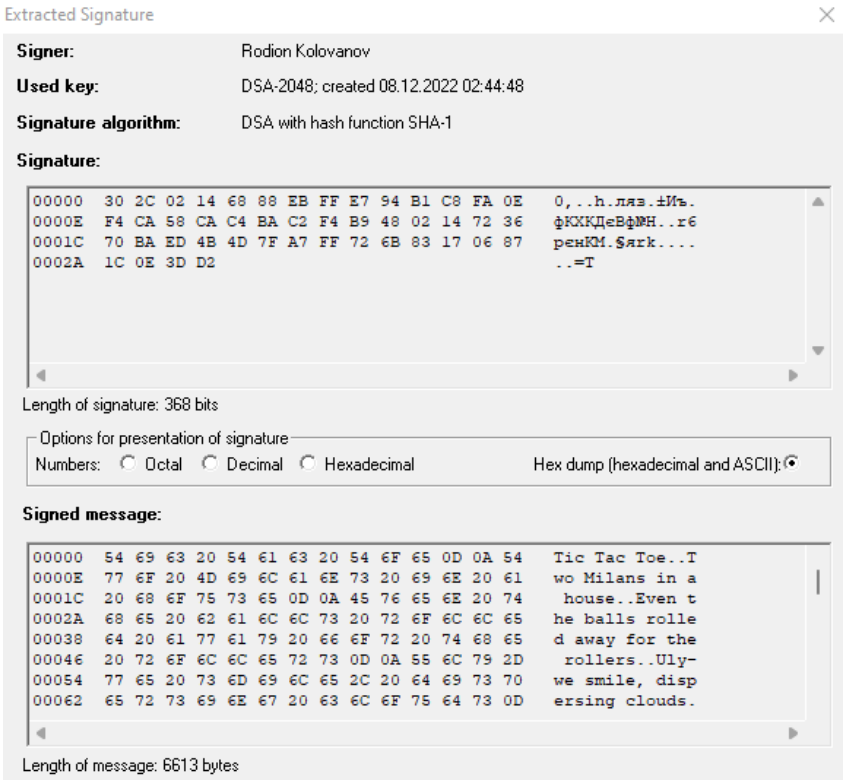


Рисунок 7 – Цифровая подпись, сгенерированная алгоритмом DSA с использованием хэш-функции SHA-1 и ключа DSA-2048.

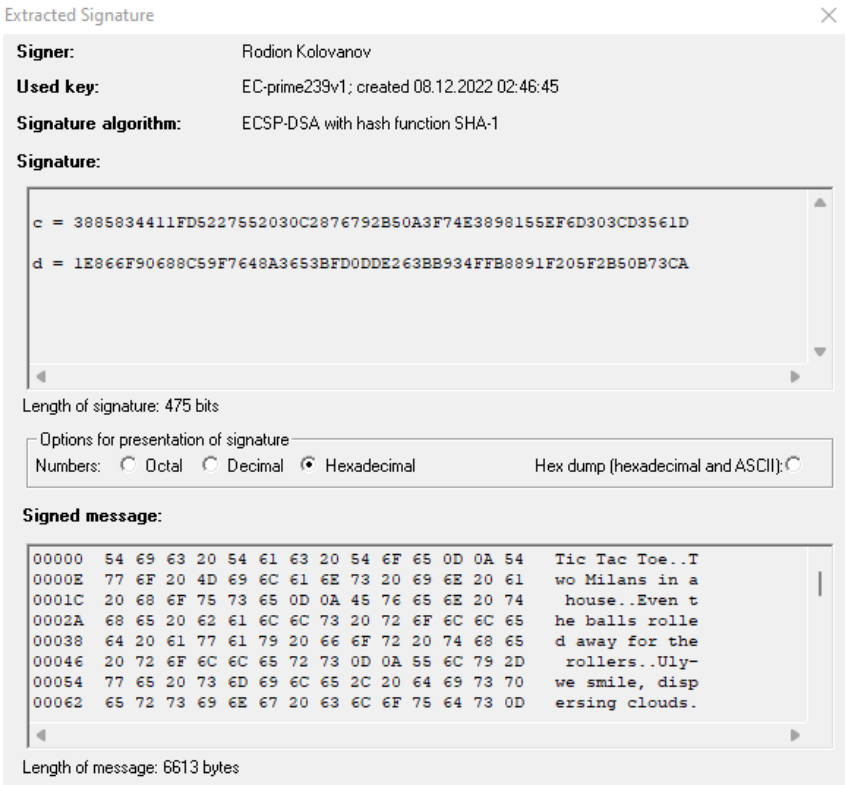


Рисунок 8 – Цифровая подпись, сгенерированная алгоритмом ECSP-DSA с использованием хэш-функции SHA-1 и ключа EC-prime239v1.

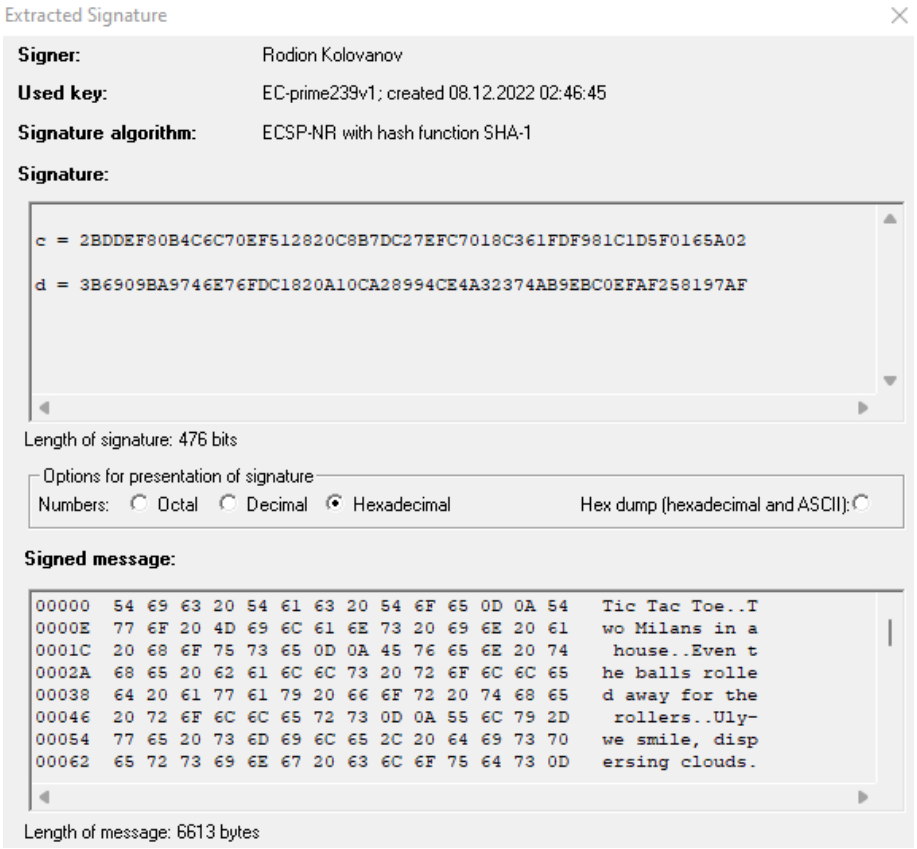


Рисунок 9 – Цифровая подпись, сгенерированная алгоритмом ECSP-NR с использованием хэш-функции SHA-1 и ключа EC-prime239v1.

Далее была выполнена процедура проверки цифровой подписи «Digital Signatures/PKI -> Verify Signature» для случаев сохранения и нарушения целостности. Результаты представлены на рисунках 10-17.

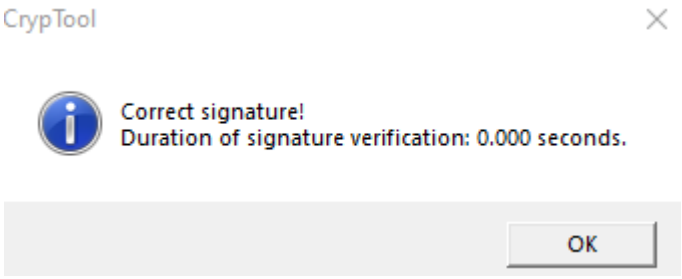


Рисунок 10 – Проверка цифровой подписи для алгоритма RSA при сохранении целостности.

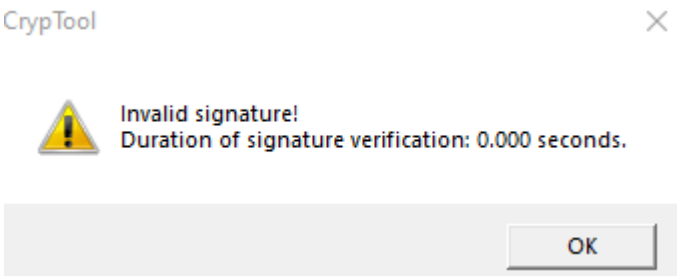


Рисунок 11 – Проверка цифровой подписи для алгоритма RSA при нарушении целостности.

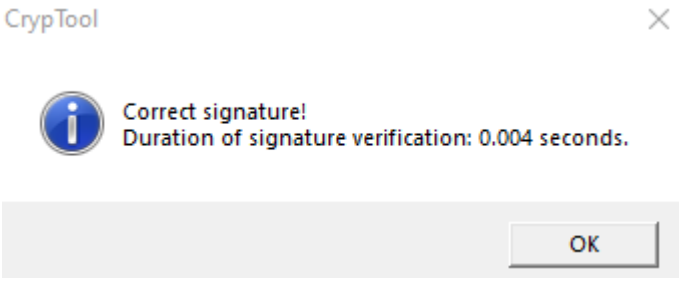


Рисунок 12 – Проверка цифровой подписи для алгоритма DSA при сохранении целостности.

Рисунок 13 – Проверка цифровой подписи для алгоритма DSA при нарушении целостности.

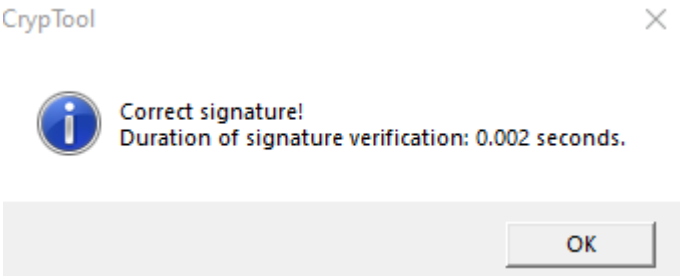


Рисунок 14 – Проверка цифровой подписи для алгоритма ECSP-DSA при сохранении целостности.

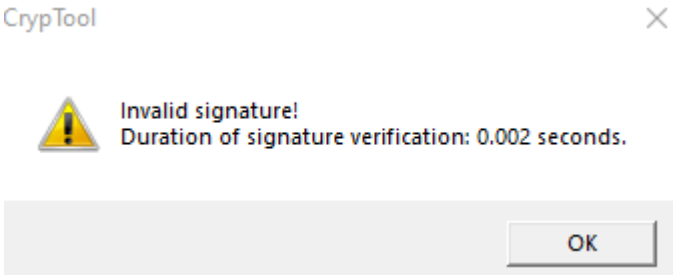


Рисунок 15 – Проверка цифровой подписи для алгоритма ECSP-DSA при нарушении целостности.

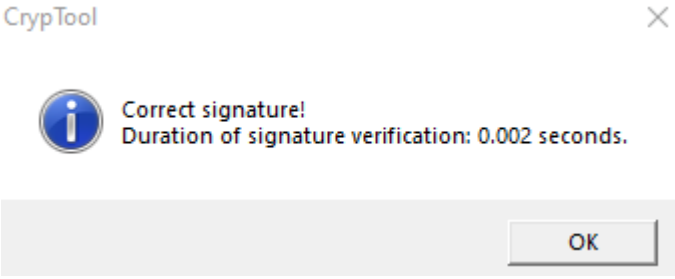


Рисунок 16 – Проверка цифровой подписи для алгоритма ECSP-NR при сохранении целостности.

Рисунок 17 – Проверка цифровой подписи для алгоритма ECSP-NR при нарушении целостности.

Схемы цифровой подписи на эллиптических кривых.

Задание.

1. Выполните процедуру создание подписи «Digital Signatures / PKI -> Sign Document» алгоритмом ECSP-DSA в пошаговом режиме (Display inter. results=ON). Зафиксируйте скриншоты последовательности шагов;

2. Выполните процедуру проверки подписи ECSP-DSA для случаев сохранения и нарушения целостности исходного текста. Сохраните скриншоты результатов;
3. Проверить лекционный материал по ECDSA, выполнив создание и проверку подписи сообщения M (принять M=h(M)) приложением «Indiv.Procedures -> Number Theory -> Point Addition on EC».

Описание алгоритма формирования и проверки подписи ECDSA.

Схема цифровой подписи ECDSA представлена на рисунке 18.

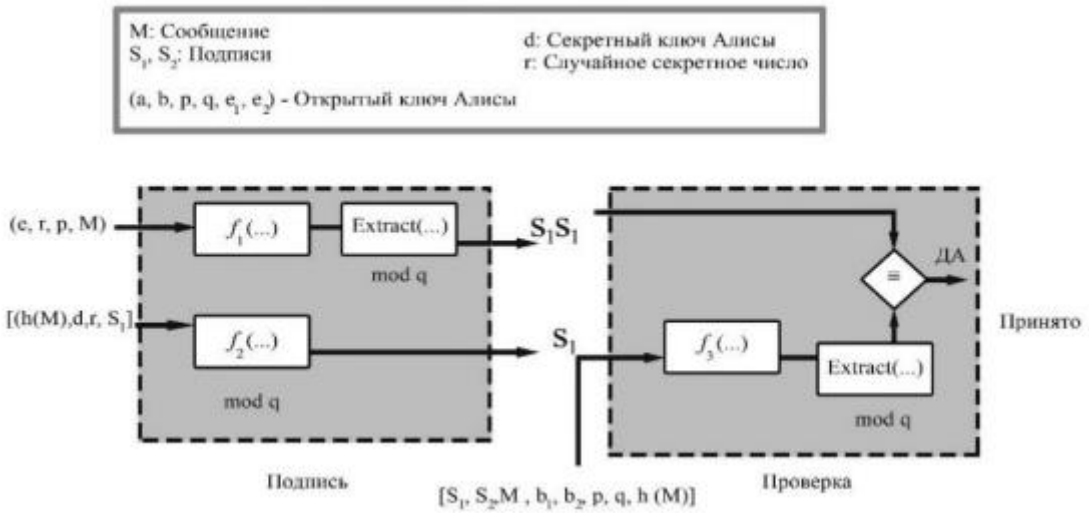


Рисунок 18 – Схема цифровой подписи ECDSA.

В процессе подписания две функции f_1 и f_2 и экстрактор *Extract* 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.

После того, как сгенерирована ключевая пара (закрытый ключ – d, и открытый ключ – (a, b, q, p, e_1, e_2), осуществляется подписание документа, затем на принимающей стороне осуществляется проверка. Схема представлена на рисунке 19.

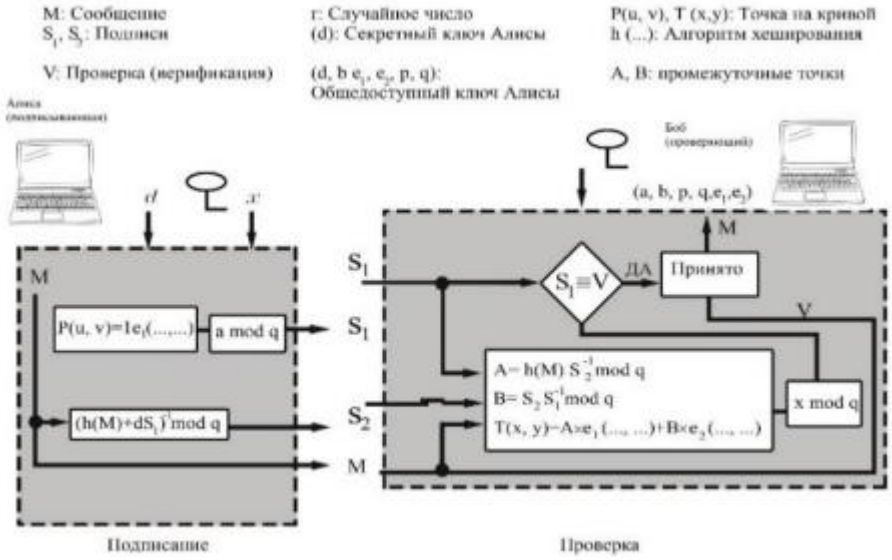


Рисунок 19 – Проверка цифровой подписи ECDSA.

Алгоритм подписания ECDSA состоит из следующих операций:

1. Выбирается секретное случайное число r: $r \in (1, q-1)$;
2. Выбирается третья точка на кривой: $P(u, v) = r \times e_1$;
3. Вычисляется первая часть подписи по формуле:

$$S_1 = u \bmod q ,$$

где u – абсцисса;

4. Вычисляется вторая часть подписи по формуле:

$$S_2 = (h(M) + d \times S_1) \times r^{-1} \bmod q ,$$

где $h(M)$ – дайджест сообщения, d – закрытый ключ.

Алгоритм проверки цифровой подписи ECDSA включает следующие операции:

1. Вычисляем промежуточные результаты A и B:

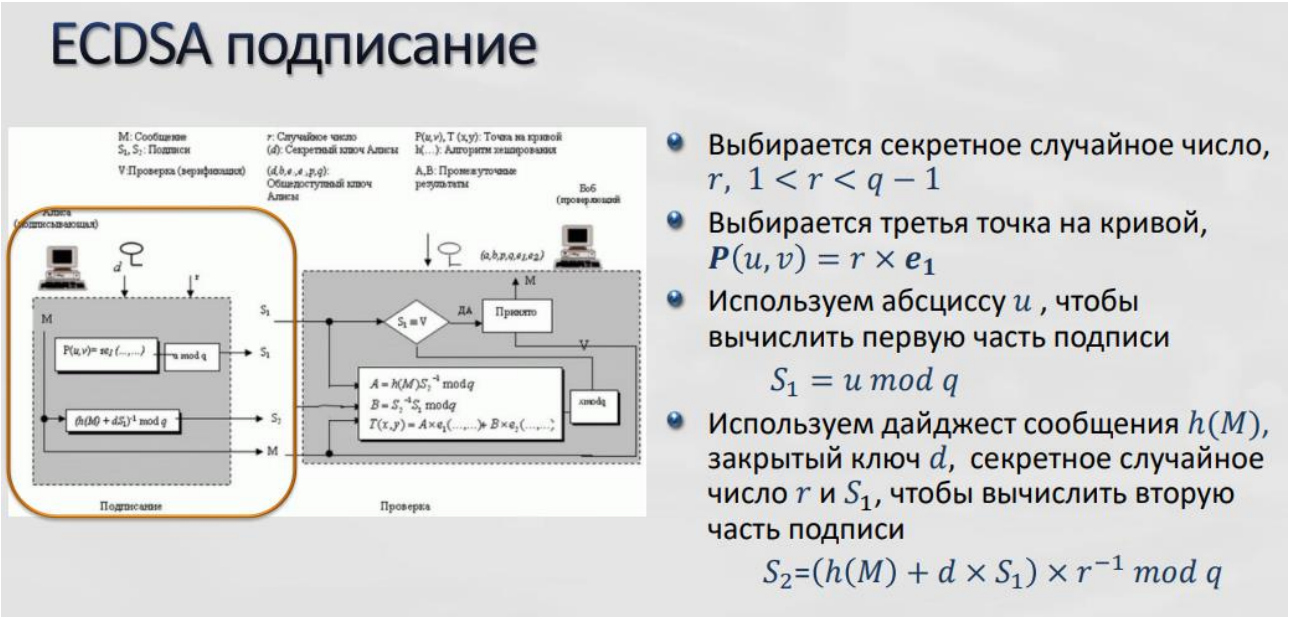
$$A = h(M) \times {S_2}^{-1} mod\ q$$

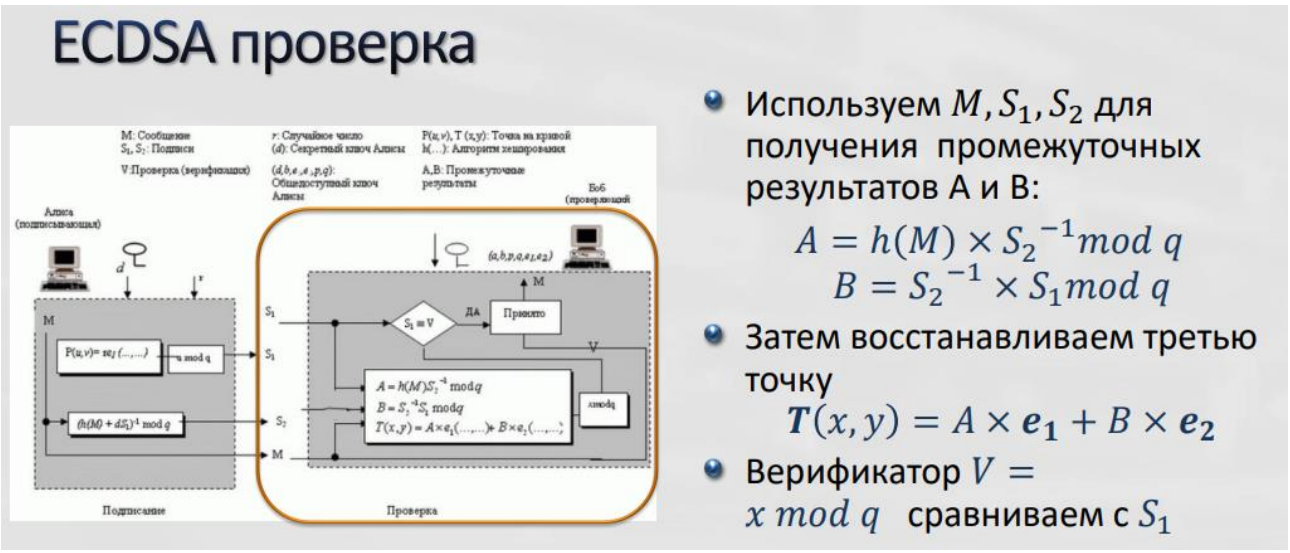
$$B = {S_2}^{-1} \times S_1 \bmod q$$

2. Восстанавливаем третью точку:

$$T(x, y) = A \times e_1 + B \times e_2$$

3. Верификатор $V = x \ mod \ q$ сравнивается с первой частью цифровой подписи S_1.





Сравним лекционную версию ECDSA и реализацию из CrypTool 1. Результаты представлены в таблице 3.

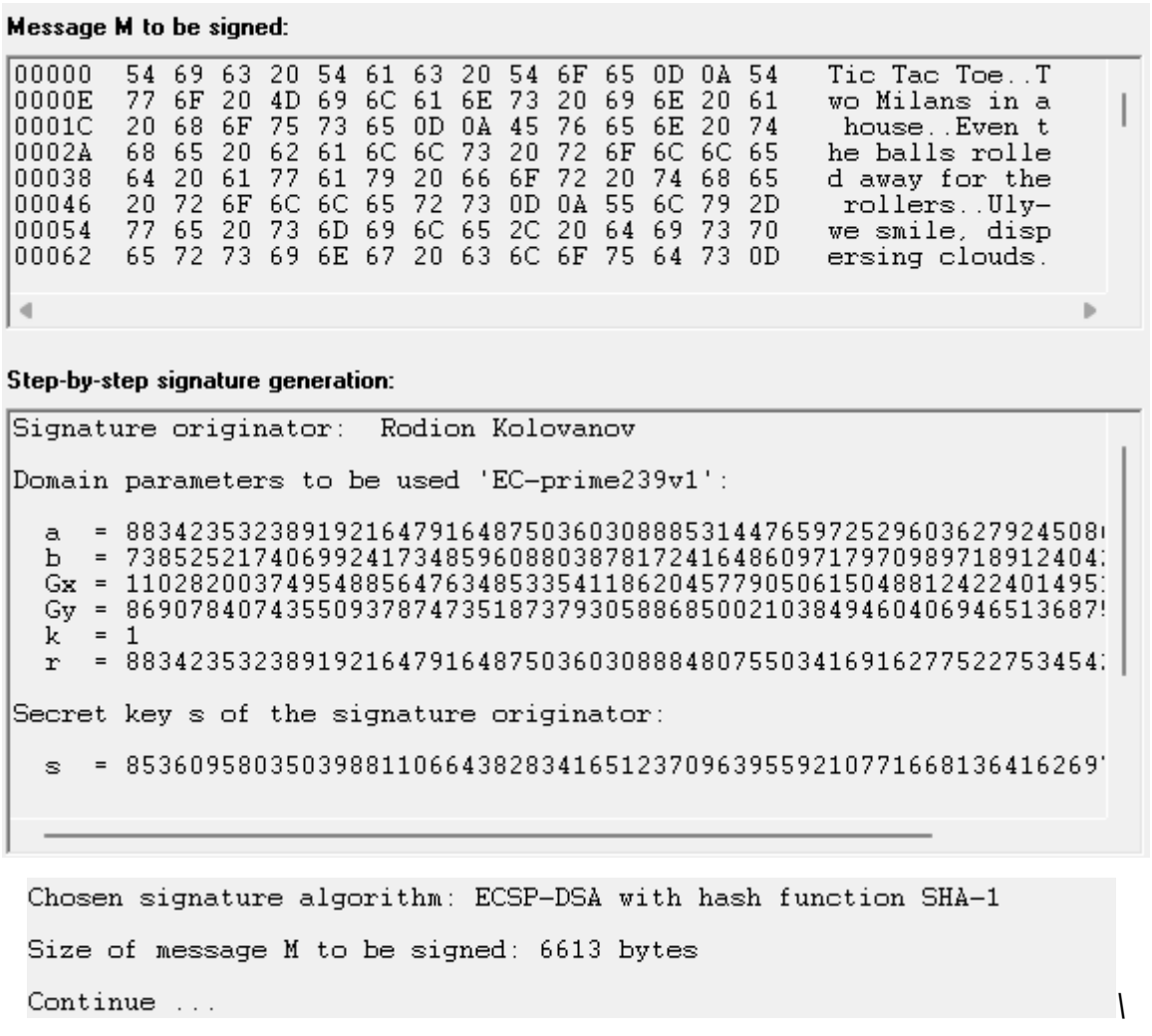
Таблица 3 – Сравнение лекционной версии ECDSA и реализации из CrypTool.

Параметр из CrypTool 1	Параметр из лекции
a	*a*
b	*b*
(Gx, Gy)	$e_1 = (x, y)$

r	*q*
s	*d*
f	*h(M)*
u	*r*
i	*u*
(Vx, Vy)	*P(u, v)*
c	S_1
d	S_2

Выполнение алгоритма формирования и проверки подписи ECDSA.

Далее при помощи утилиты «Digital Signatures / PKI -> Sign Document» была создана подпись алгоритмом ECSP-DSA в пошаговом режиме. Последовательность шагов представлена на следующих рисунках:



```
Calculate a 'hash value' f (message representative) from message M,

  f   = 432820510414143238207426353165903417123530766731

Continue ...
```

```
Create a random one-time key pair (secret key, public key) = (u,V)
with the domain parameters of 'EC-prime239v1' (V=(Vx,Vy) is a point

  u   = 325831063967745835605554102491871840602723999071026826657240
  Vx  = 880085608854081000737376978814151523871986495396000952625086
  Vy  = 642318763344493487781464636970305088337553276475839389174943

Continue ...
```

```
Convert the group element Vx (x co-ordinates of point V on elliptic

  i   = 8800856088540810007373769788141515238719864953960009526250864

Continue ...
```

```
Calculate the number c = i mod r (c not equal to 0):

  c   = 8800856088540810007373769788141515238719864953960009526250864

Continue ...
```

```
Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):

  d   = 847227543406582362457458072961742136408454816292964401683078

Continue ...
```

```
Signature generation finished.
The signature consists of the two numbers c and d.
```

Далее при помощи утилиты «Digital Signatures/PKI -> Verify Signature» была выполнена проверка подписи для случаев сохранения и нарушения целостности. Результаты представлены на рисунках 20 и 21.

Рисунок 20 – Проверка цифровой подписи для алгоритма ECSP-DSA при сохранении целостности.

Рисунок 21 – Проверка цифровой подписи для алгоритма ECSP-DSA при нарушении целостности.

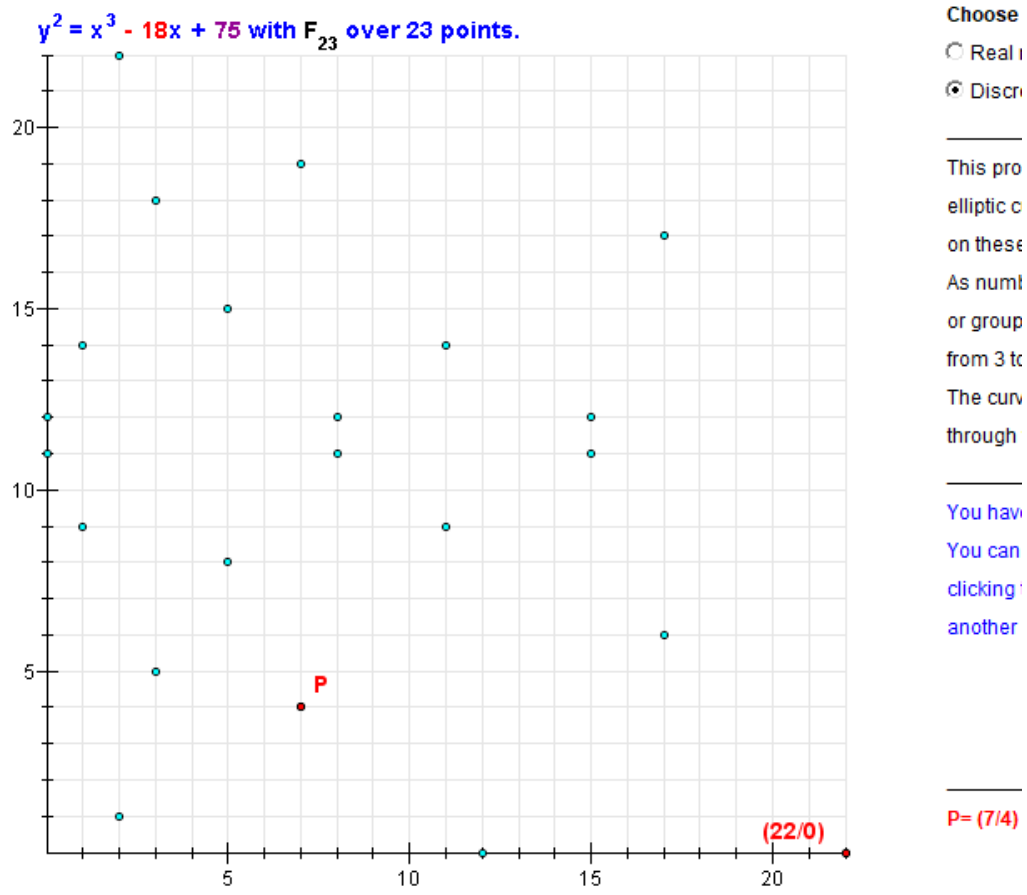
Далее были выполнены создание и проверка подписи сообщения $M = h(M)$ утилитой «Indiv.Procedures -> Number Theory -> Point Addition on EC».

1) Генерация ключей.

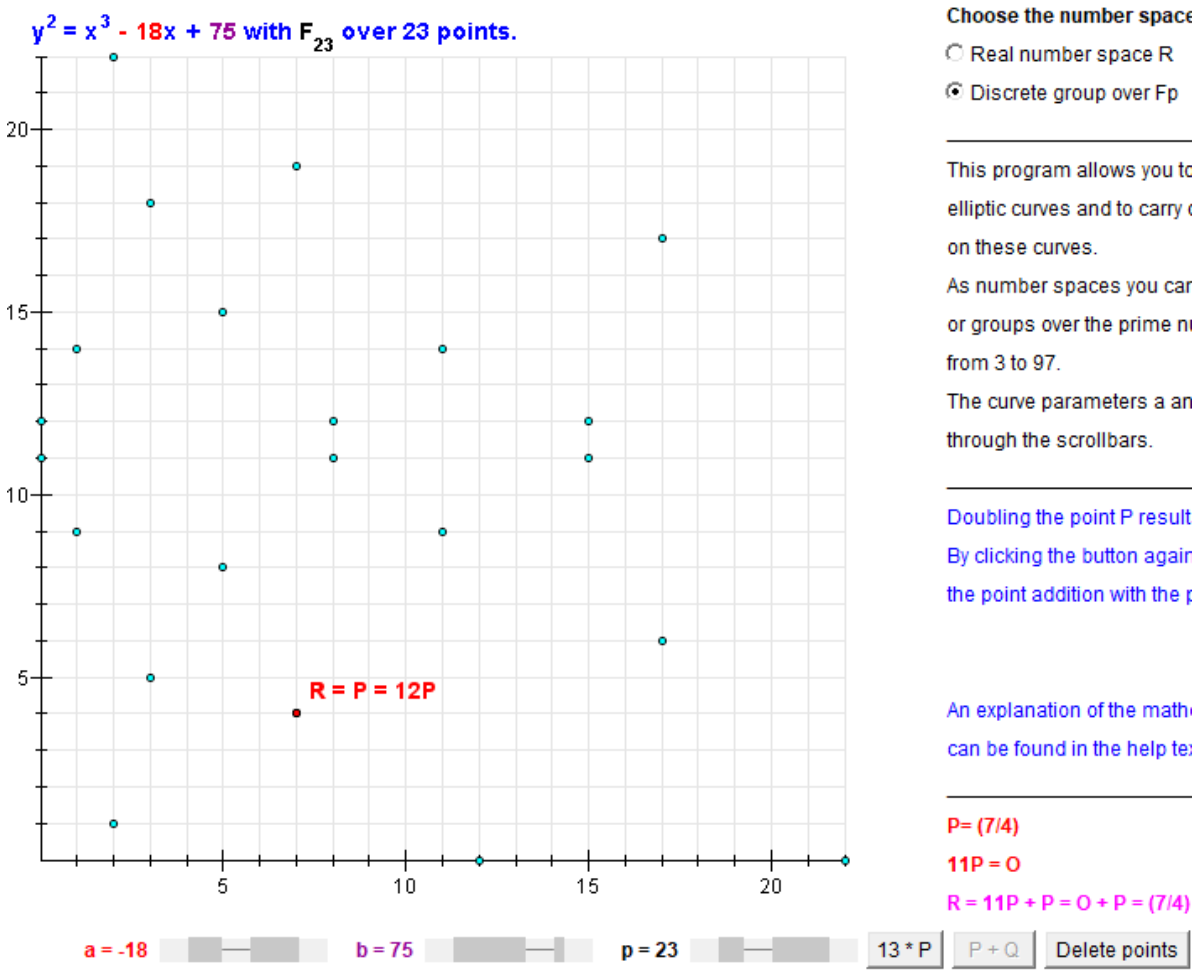
Были взяты следующие параметры эллиптической кривой:

$$a = -18,\ b = 75,\ p = 23.$$

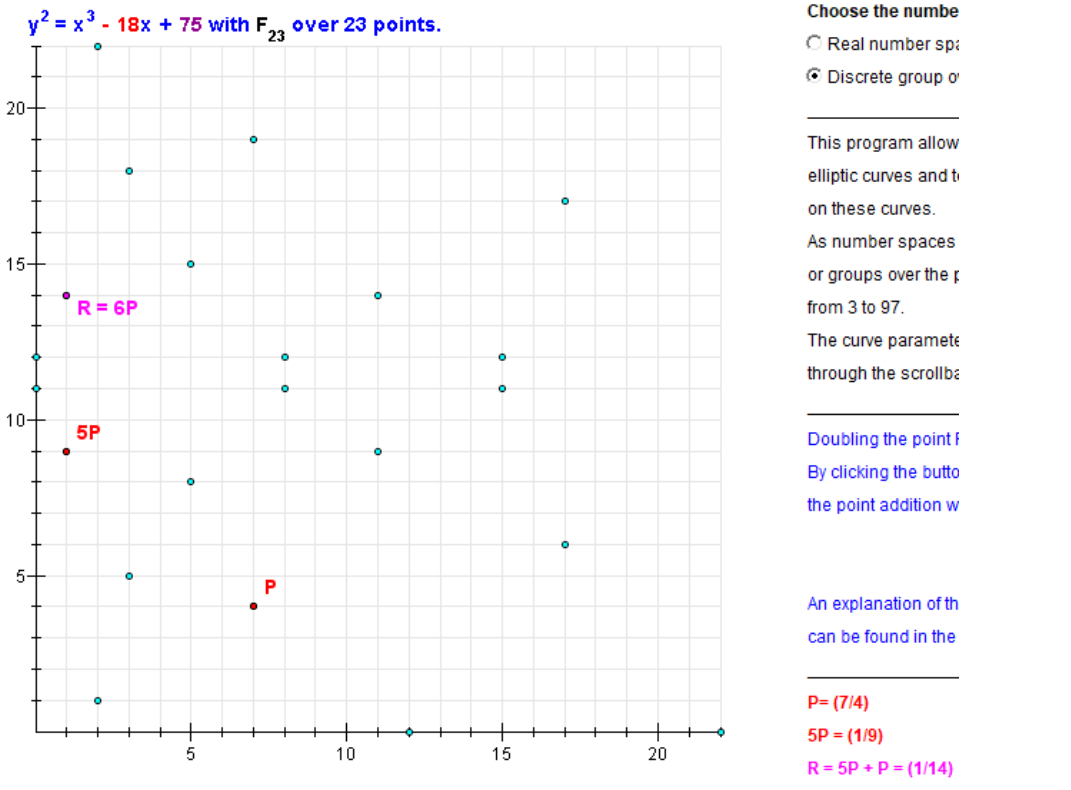
Далее была выбрана точка на кривой $P = e_1 = (7, 4)$.



Далее было подобрано такое число q, что $q \times (x_1, y_1) = O$: $q = 11$.



Далее было выбрано целое число d ($1 < d < q - 1 = 10$): d = 5 – закрытый ключ, после чего была вычислена точка $e_2 = d \times e_1 = (1, 9)$.



Открытый ключ получен:

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

2) Подписание.

Выбираем секретное случайное число r = 3. Далее выбираем третью точку на кривой $P = r \times e_1 = (15, 11)$. Далее, используя абсциссу $u = 15$, вычисляем первую часть подписи $S_1 = u \bmod q = 15 \bmod 11 = 4$.

Пусть исходное сообщение $M = h(M) = 66$. Тогда теперь можно вычислить вторую часть подписи $S_2 = (h(M) + d \times S_1) \times r^{-1} \bmod q = (66 + 5 \times 4) \times 4 \bmod 11 = 344 \bmod 11 = 3$.

Итого получаем (M = 66, $h(M)$ = 66, S_1 = 4, S_2 = 3).

3) Проверка подписи.

Вычислим промежуточные результаты A и B:

$A = h(M) \times S_2^{-1} \bmod q = 66 \times 4 \bmod 11 = 0$.

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

Далее восстанавливаем третью точку T:

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

Далее находим верификатор V:

$V = x \bmod q = 15 \bmod 11 = 4$.

Сравниваем верификатор V = 4 с S_1 = 4 – они равны, значит подпись корректна.

Демонстрация процесса подписи в среде PKI.

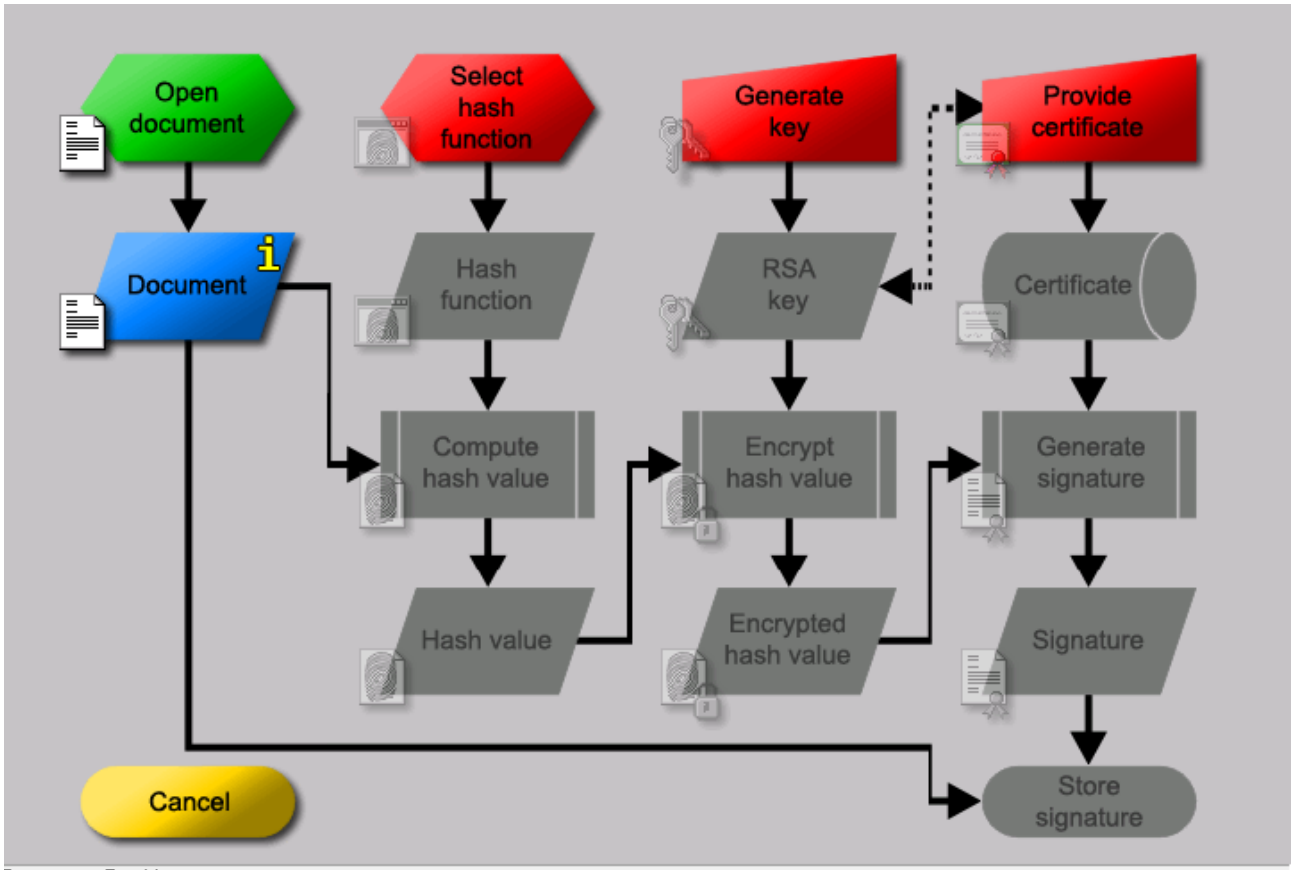
Задание.

1. Запустить демонстрационную утилиту «Digital Signatures / PKI -> Signature Demonstration»;
2. Получите сертификат на ранее сгенерированную ключевую пару RSA-2048;

3. Выполните и сохраните скриншоты всех этапов создания цифровой подписи документа;
4. Сохраните скриншот сертификата для проверки этой цифровой подписи.

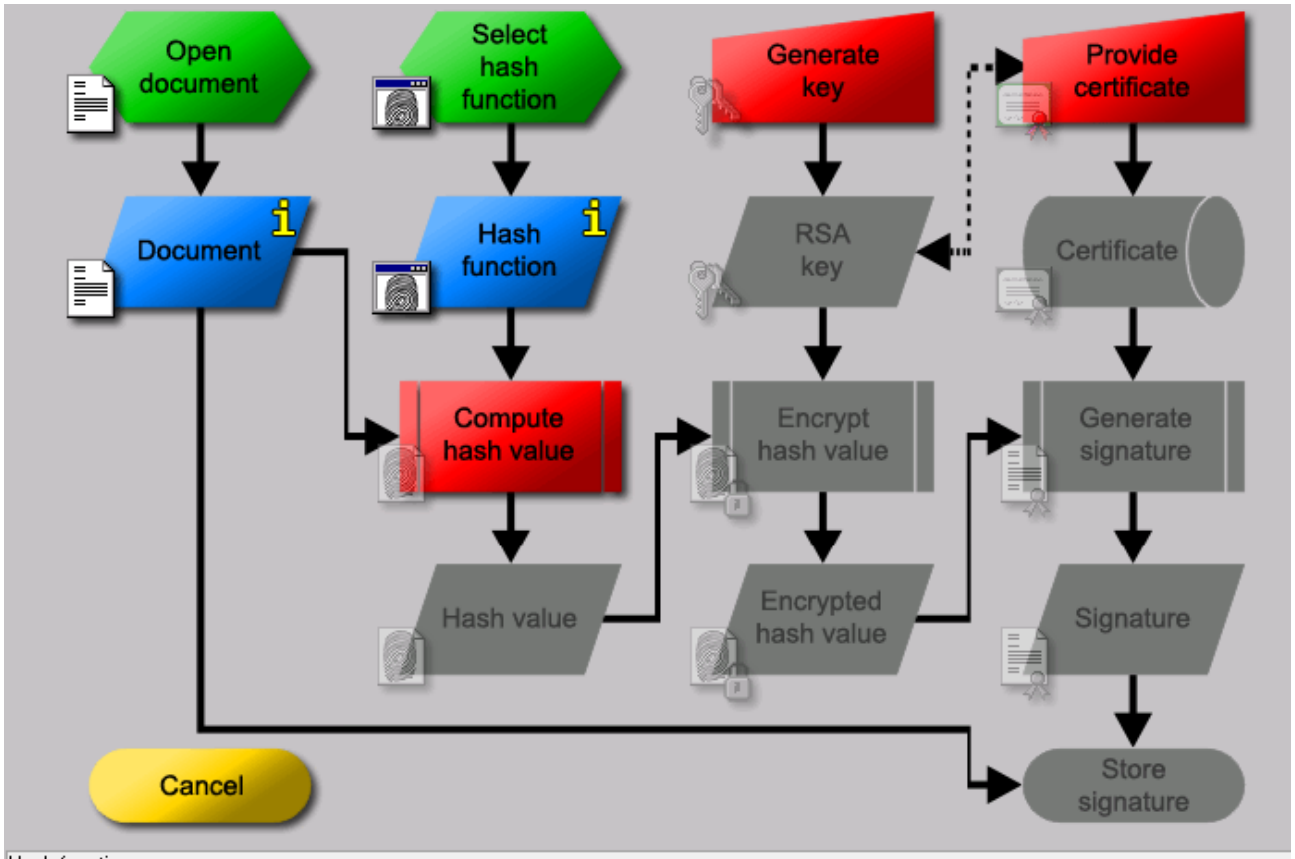
Схема процедуры подписания из CrypTool.

Была запущена демонстрационная утилита «Digital Signatures / PKI -> Signature Demonstration». При помощи утилиты получим сертификат для ранее сгенерированной ключевой пары RSA-2048.



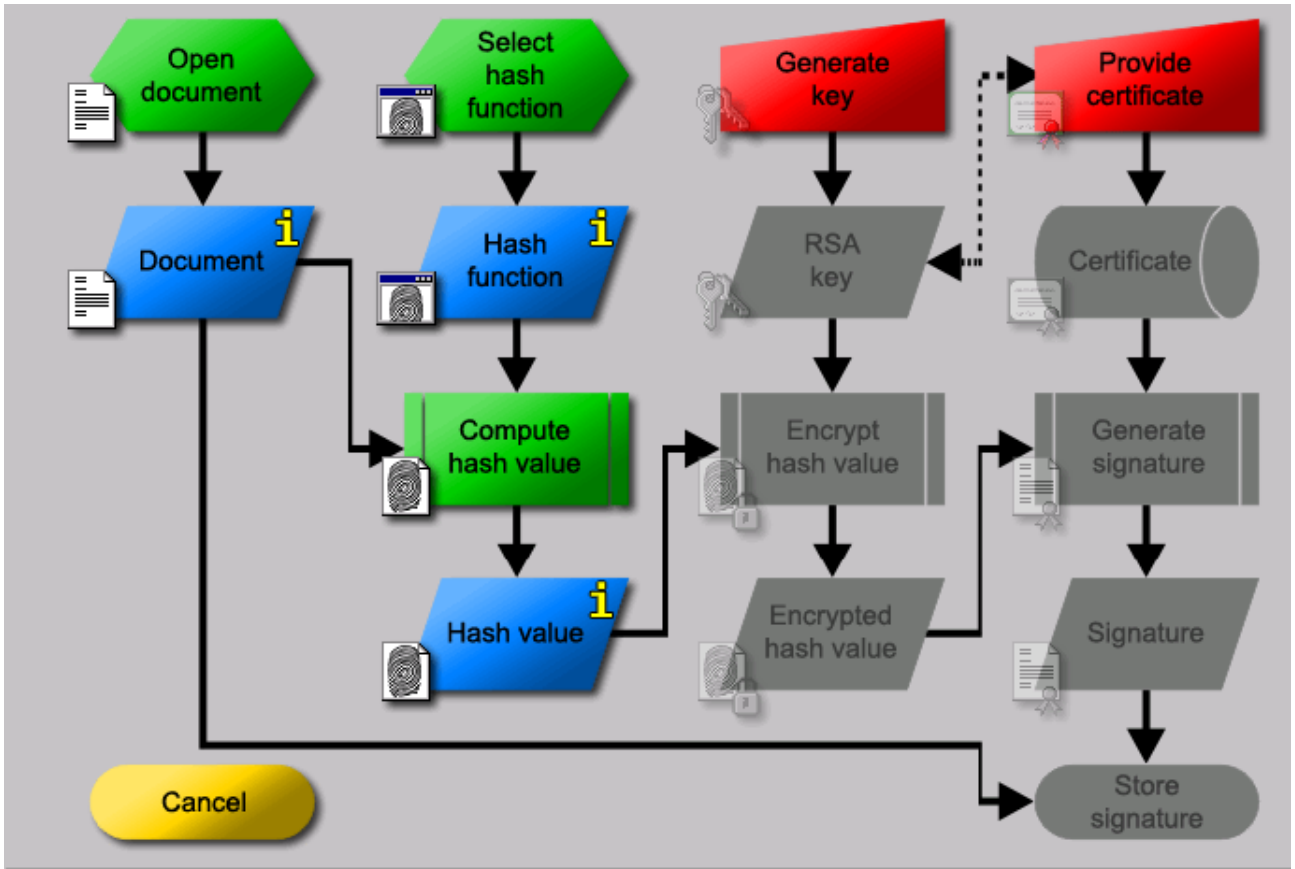
Document: Две Миланы в домике.txt

```
00000   54 69 63 20 54 61 63 20 54 6F 65 0D 0A 54 77 6F 20 4D   Tic Tac Toe..Two M
00012   69 6C 61 6E 73 20 69 6E 20 61 20 68 6F 75 73 65 0D 0A   ilans in a house..
00024   45 76 65 6E 20 74 68 65 20 62 61 6C 6C 73 20 72 6F 6C   Even the balls rol
00036   6C 65 64 20 61 77 61 79 20 66 6F 72 20 74 68 65 20 72   led away for the r
00048   6F 6C 6C 65 72 73 0D 0A 55 6C 79 2D 77 65 20 73 6D 69   ollers..Uly-we smi
0005A   6C 65 2C 20 64 69 73 70 65 72 73 69 6E 67 20 63 6C 6F   le, dispersing clo
0006C   75 64 73 0D 0A 50 65 6F 70 6C 65 20 63 61 6C 6C 20 75   uds..People call u
0007E   73 20 6E 61 6D 65 73 20 74 68 61 74 20 77 65 20 61 72   s names that we ar
00090   65 20 74 77 6F 20 77 68 79 0D 0A 0D 0A 4C 65 74 20 67   e two why....Let g
000A2   6F 2C 20 6E 6F 74 20 74 68 65 20 77 61 79 0D 0A 42 75   o, not the way..Bu
000B4   74 20 77 68 6F 20 61 6D 20 49 20 74 65 6C 6C 69 6E 67   t who am I telling
```



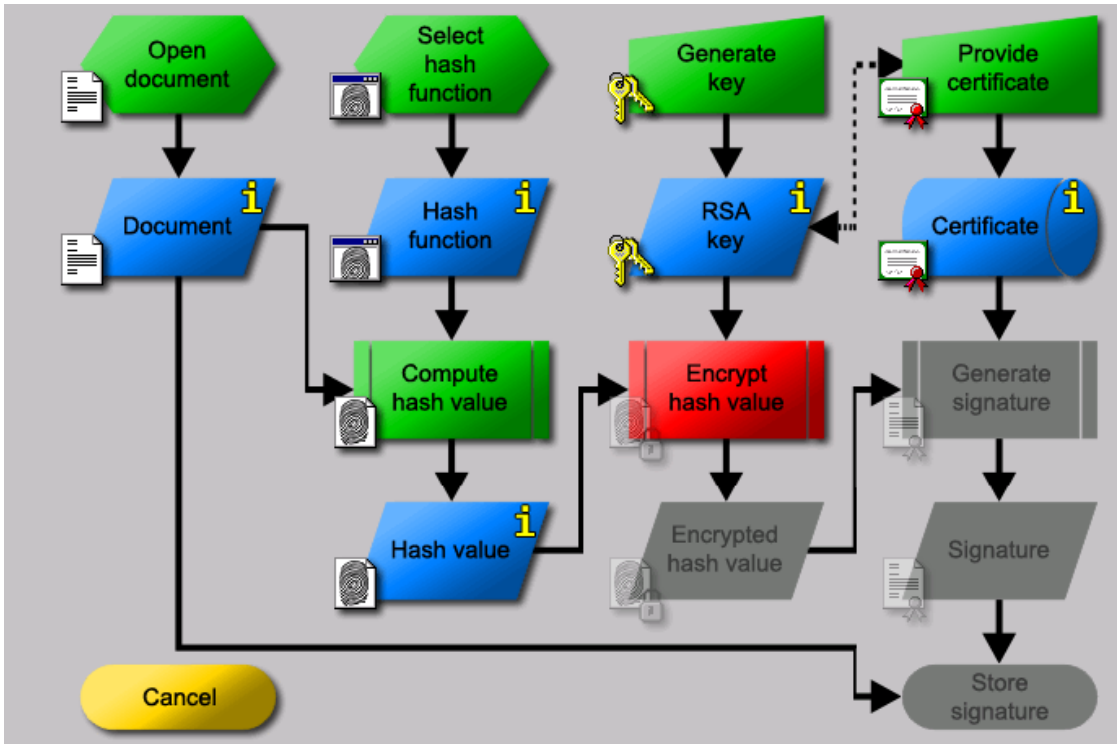
Hash function

```
Name:                   SHA-1
Length in bits:         160
Algorithm ID:           30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14
```



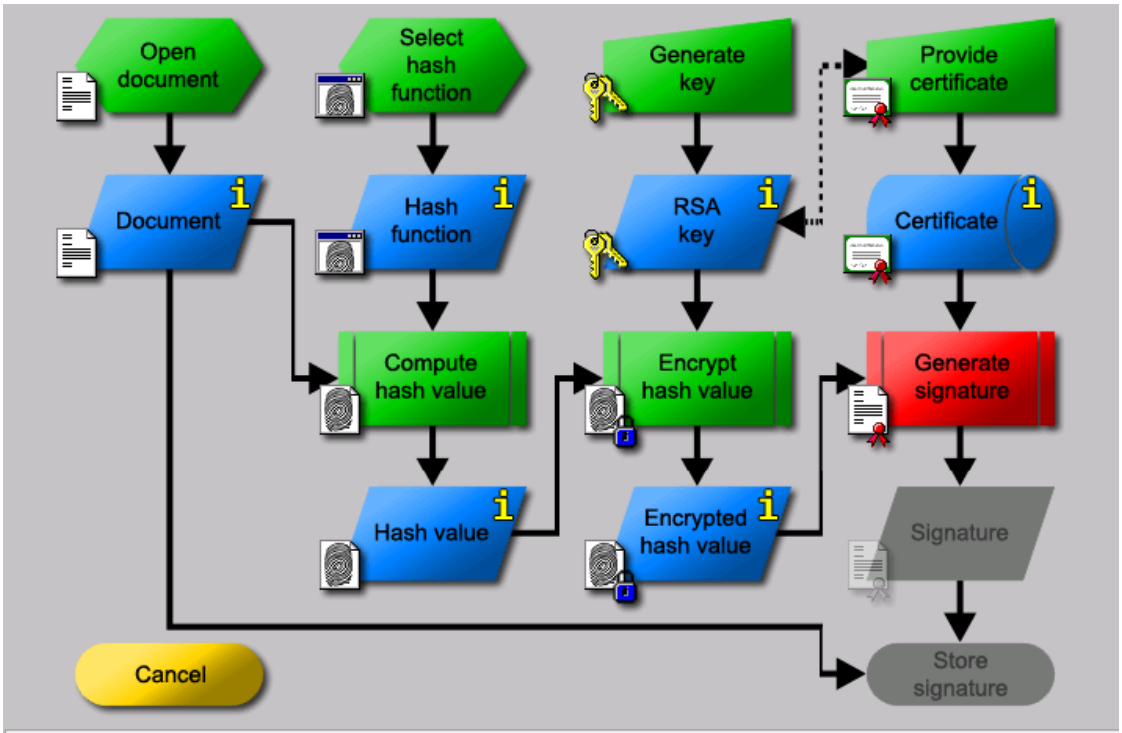
SHA-1 hash of <Две Миланы в домике.txt>

```
4B D0 57 CA 47 94 97 A1 E2 7E AE 2C 7E 71 92 A4 3C 22 79 8B
```



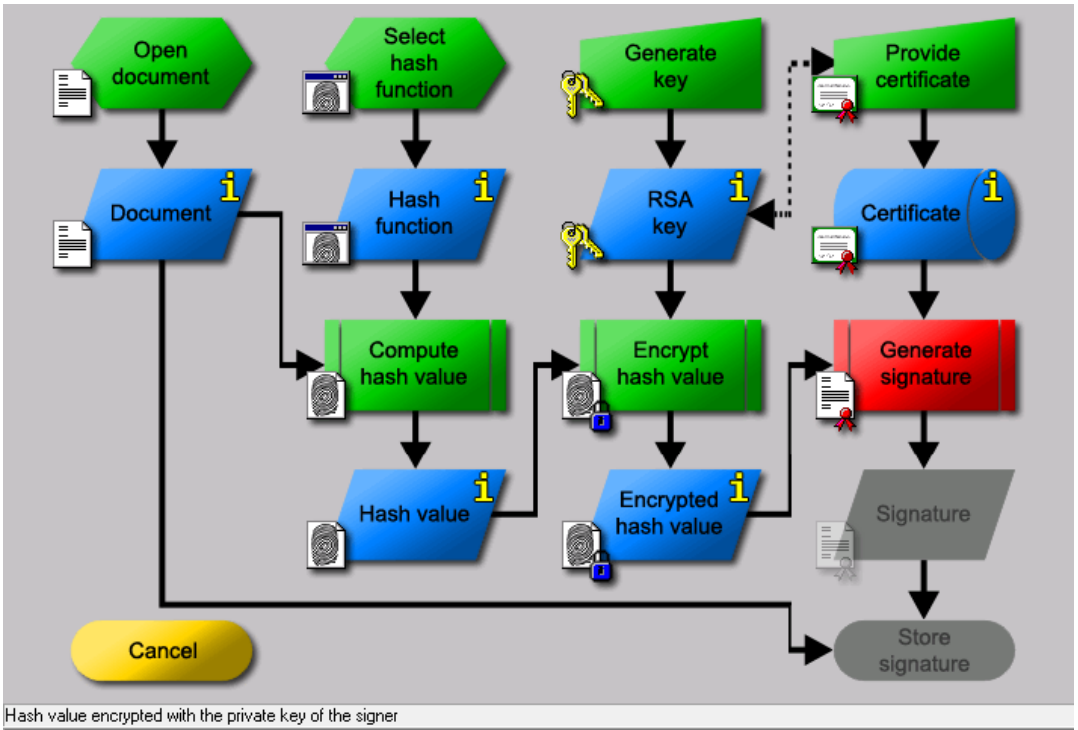
Certificate of signer (added to signature for examination)

```
Version:                    2 (X.509v3-1996)
SubjectName:                CN=Rodion Kolovanov [1670456581], DC=cryptool, DC=org
IssuerName:                 CN=CrypTool CA 2, DC=cryptool, DC=org
SerialNumber:               C8:54:DE:71:1F:B1:F6:0E
Validity  -  NotBefore:     Thu Dec 08 02:43:01 2022 (221207234301Z)
             NotAfter:      Fri Dec 08 02:43:01 2023 (231207234301Z)
Public Key Fingerprint:     C933 4D74 4C84 579A 61E0 2473 2104 6CF4
SubjectKey:                 Algorithm rsa (OID 2.5.8.1.1), Keysize = 2048
            Public modulus (no. of bits = 2048):
              0  FFC1ED5E DC83F0E0  4C280059 00CC5017
             10  3E94EBD9 02D63306  CAB7AC37 89E592F4
```



RSA key of signer

```
Bit length of N:          2048
RSA modulus N:            322863968469713824958075448077175582061121624945962628126889783
phi(N) = (p-1)(q-1):      322863968469713824958075448077175582061121624945962628126889783
Public key:               65537
Private key:              287777876896384227000411634640404834308870099656963347758247504
```

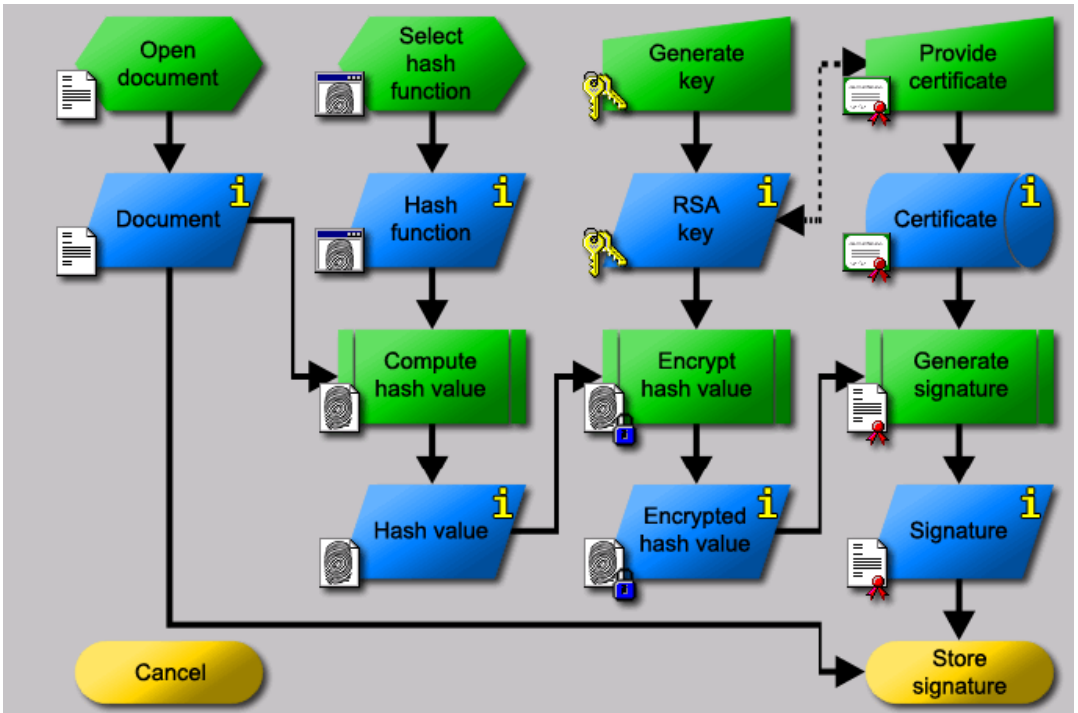


Hash value encrypted with the private key of the signer

```
Padding string:          01 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF I
Algorithm ID:            30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14
Hash value:              4B D0 57 CA 47 94 97 A1 E2 7E AE 2C 7E 71 92 A4 3C 22 79 8B

ASN-1 hash value:        01 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF I
Length in bits:          2040

Encrypted hash value:    DE B2 A8 CB D8 D6 84 2A 0C AB EC 32 84 C9 58 E6 77 AB 56 51 FB (
Length in bits:          2048
```



SHA-1 signature of <Две Миланы в домике.txt>

```
00000   53 69 67 6E 61 74 75 72 65 3A 20 20 20 20 20 20 20 DE   Signature:       Þ
00012   B2 A8 CB D8 D6 84 2A 0C AB EC 32 84 C9 58 E6 77 AB 56   І¨ЛШЦ.*.«м2.ЙXжw«V
00024   51 FB 0A 90 C8 CF D9 CA 19 6D 7D 02 54 9E 18 53 2E AA   Qы..ИПЩК.m}.T..S.Є
00036   67 D8 DA BD F2 90 51 95 F7 82 69 AE 1C AC 80 A8 FD 77   gШЪSm.Q.ч.i®.¬.Ёэw
00048   C4 EA B3 5A 28 7F 4B 4E CD A0 C3 3B 90 8F 8B 38 08 4E   ДкіZ(.KNH Г;...8.N
0005A   FF 51 3F 69 ED 13 64 65 C6 C2 71 BB 2E 02 03 E9 8D 6B   яQ?iн.deЖВq».....ú.k
0006C   CC E3 A3 B2 EE D6 E6 36 51 AF BD C7 FA 66 49 1F EE 41   МгЈІоЦж6QЇЅЗъfI.оА
0007E   1F 2B E4 DA 97 AA 68 F3 07 40 8F C5 FF 0E 39 2A A7 F1   .+gЪ.Єhy.@.Ея.9*Sc
00090   8E 90 D6 C8 52 B7 7D A2 91 0F 8F AD 5A BA 87 05 0C 83   ..ЦИR·}ў....-Zе....
000A2   46 2A 2C 29 6B F1 53 AB 6D EA 93 5C EB 50 52 CD C3 41   F*,)kcS«mк.\лPRНГА
000B4   92 5F 3C 83 C8 C0 FE 88 32 46 FF 66 6D 90 C9 33 41 AA   ._<.ИАю.2Fяfm.Й3Ає
```

Сравнение структуры сертификата из лекции и сертификата из CrypTool.

Полученный на предыдущем шаге сертификат, а также его структура представлены на рисунке 22.

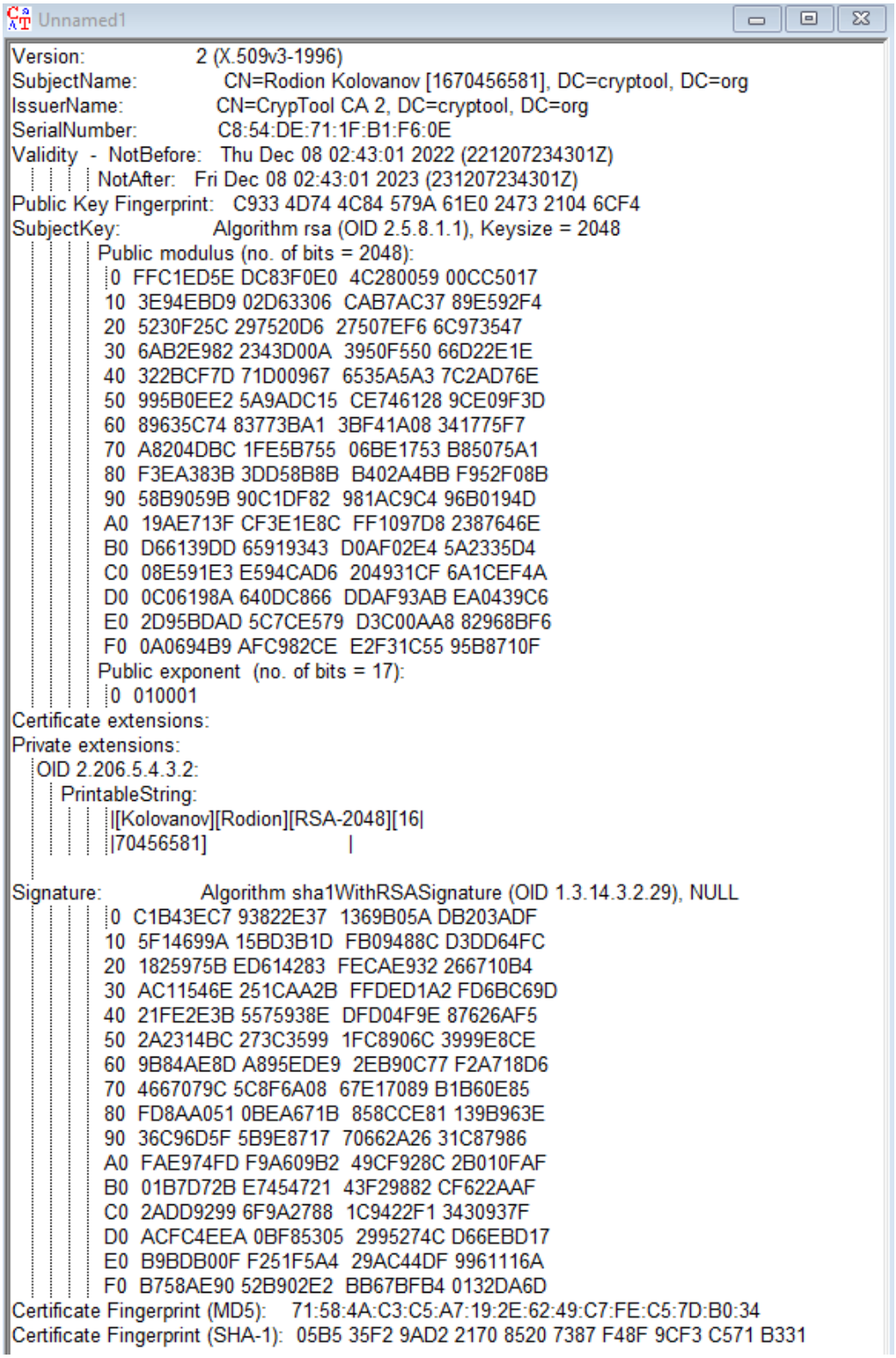


Рисунок 22 – Сертификат из CrypTool 1.

Структура сертификата из лекций представлена на рисунке 23.

Рисунок 23 – Структура сертификата из лекций.

Результаты сравнения двух сертификатов:

- Версия присутствует в обоих сертификатах;
- Серийный номер присутствует в обоих сертификатах;
- Идентификатор алгоритма подписи присутствует в обоих сертификатах;
- Имя издателя и субъекта присутствуют в обоих сертификатах;
- Период действия присутствует в обоих сертификатах;
- Информация об открытом ключе присутствуют в обоих сертификатах;
- Поле с хэшом открытого ключа есть только в версии сертификата CrypTool;
- Уникальные идентификаторы издателя и субъекта присутствуют только в версии сертификата из лекций;
- Дополнения присутствуют в обоих сертификатах;
- Подпись присутствует в обоих сертификатах;
- Поля с хэшами SHA-1 и MD5 сертификата есть только в версии сертификата CrypTool;

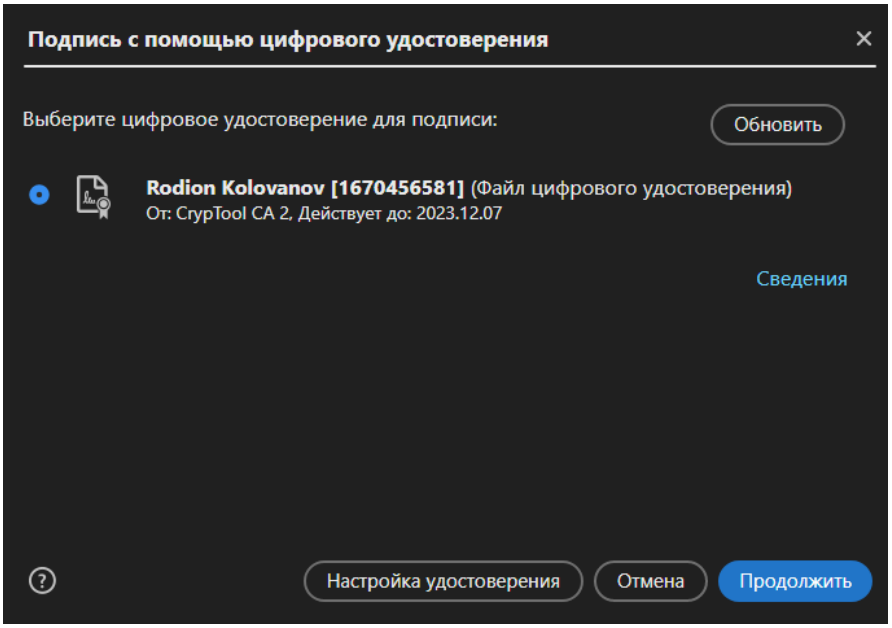
Подписание своего отчета.

Задание.

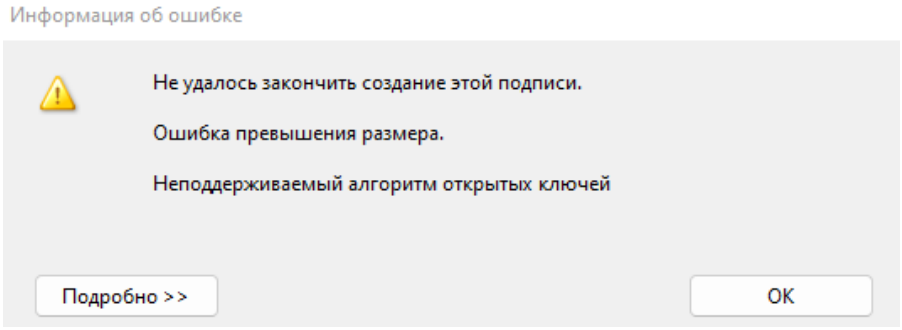
1. Сконвертируйте отчет в формат pdf;
2. Экспортируйте ранее созданный сертификат ключевой пары RSA «Digital Signatures / PKI -> PKI -> Display / Export Keys -> Export PSE(#PKCS12)»;
3. Откройте pdf-версию отчета и попытайтесь подписать с использованием этого сертификата;
4. Создайте собственный самоподписанный сертификат в среде Adobe Reader и используйте его для подписи отчета;
5. Сохраните скриншоты свойств подписи и сертификата;
6. Внесите изменения (маркеры, комментарии) в отчет и проверьте подпись.

Подписание своего отчета.

Для начала текущее состояние отчета было сконвертировано в формат pdf. Далее при помощи утилиты «Digital Signatures / PKI -> PKI -> Display / Export Keys -> Export PSE(#PKCS12)» был экспортирован ранее созданный сертификат ключевой пары RSA. Далее через программу Adobe Reader была осуществлена попытка подписать сохраненный pdf.



В результате, подписать сохраненный pdf не удалось.



Далее в среде Adobe Reader был создан собственный самоподписанный сертификат, который в дальнейшем был использован для повторного подписания сохраненного pdf. Свойства сертификата представлены на рисунке 24.

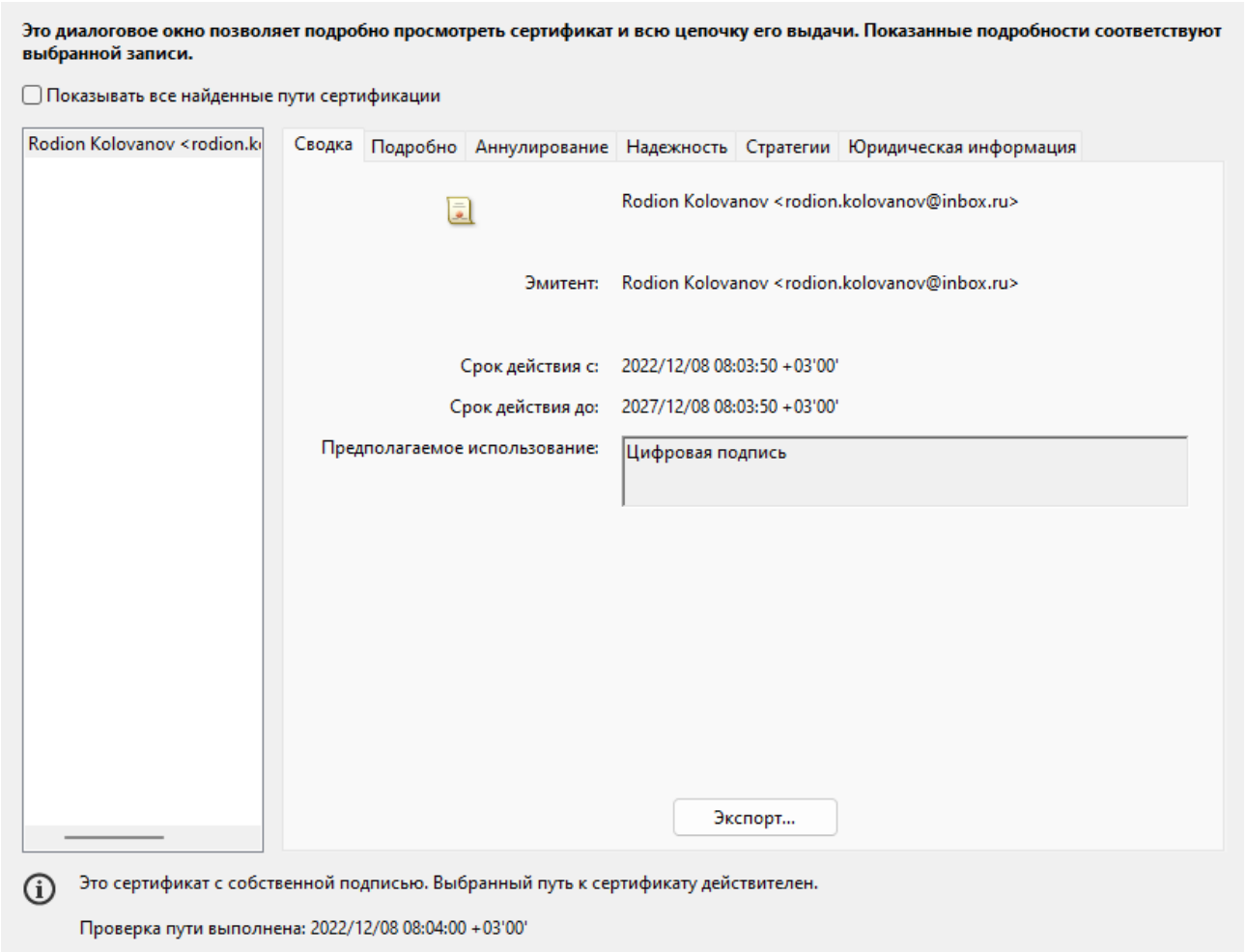


Рисунок 24 – Свойства созданного сертификата.

Далее pdf-файл был подписан при помощи созданного сертификата. Подпись и ее свойства представлены на рисунках 25, 26 и 27.

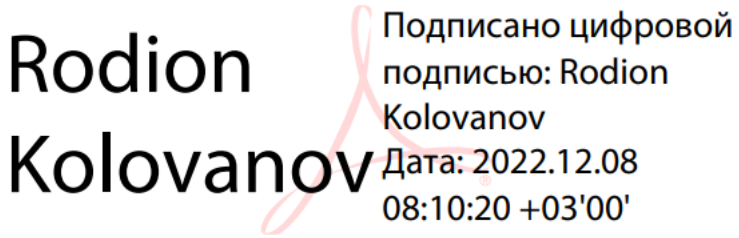


Рисунок 25 – Подпись pdf-файла.

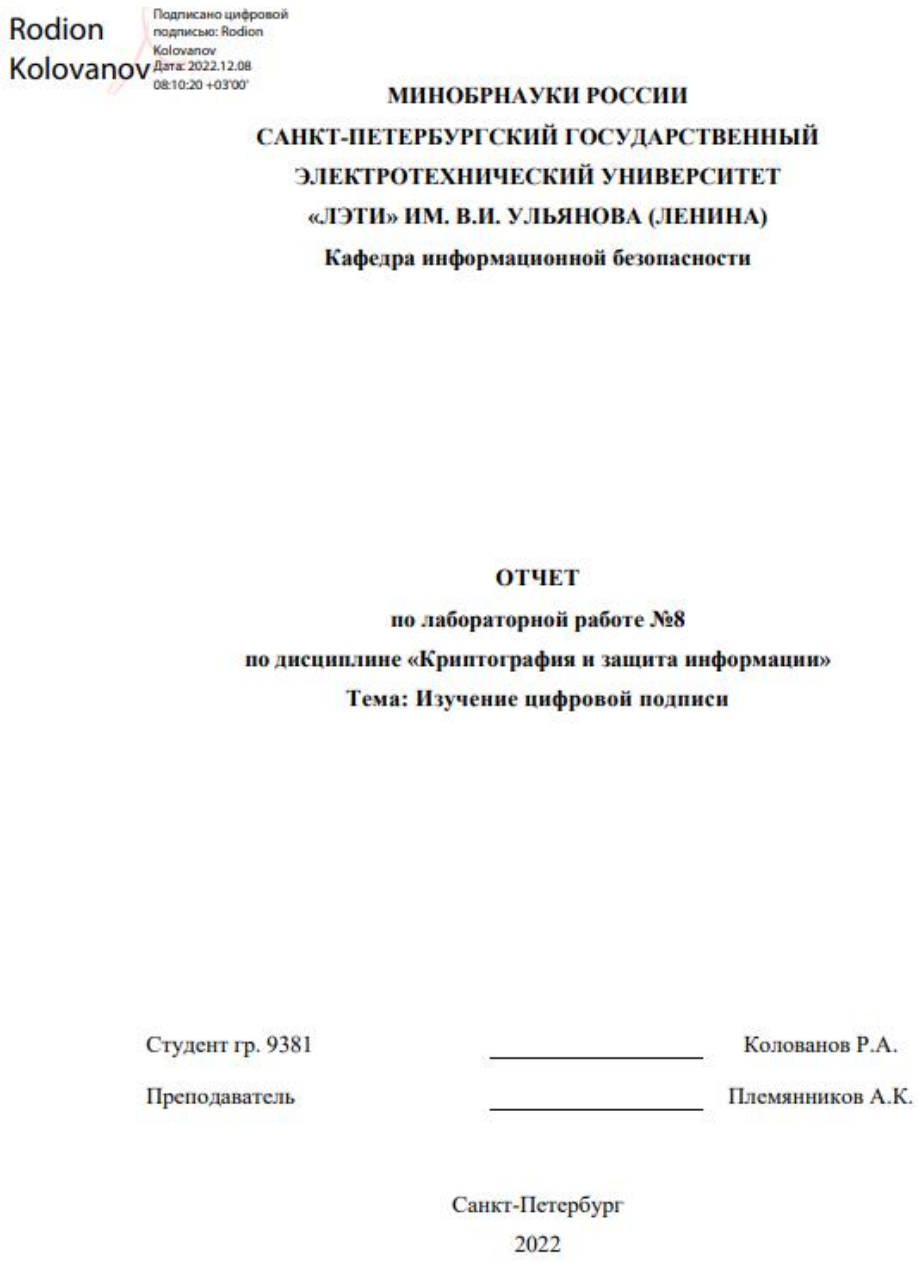


Рисунок 26 – Титульный лист pdf-файла с подписью.

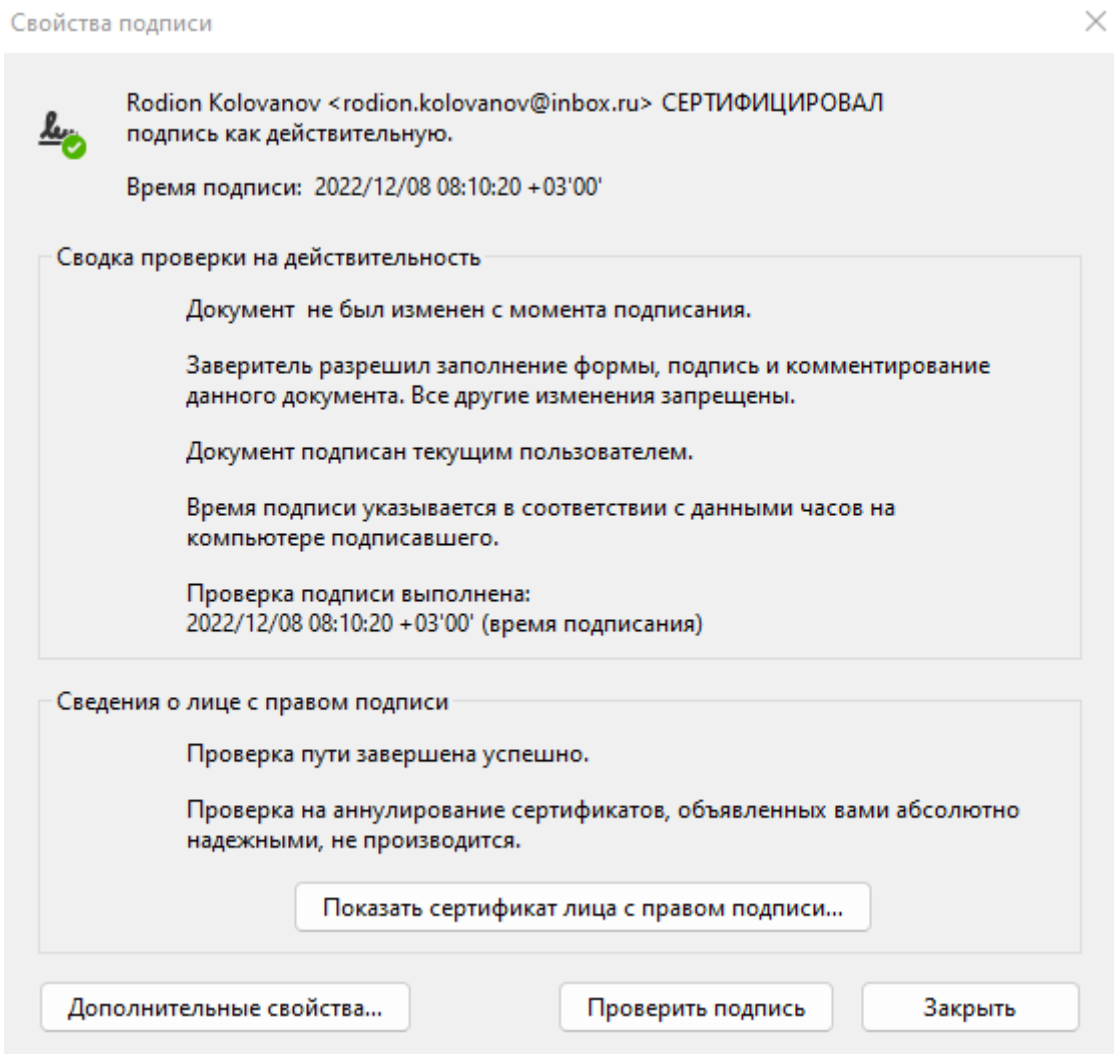


Рисунок 26 – Свойства подписи pdf-файла.

Далее в pdf-файл были внесены изменения (добавлены маркеры и комментарии), после чего подпись файла была проверена. Результат представлен на рисунке 28.

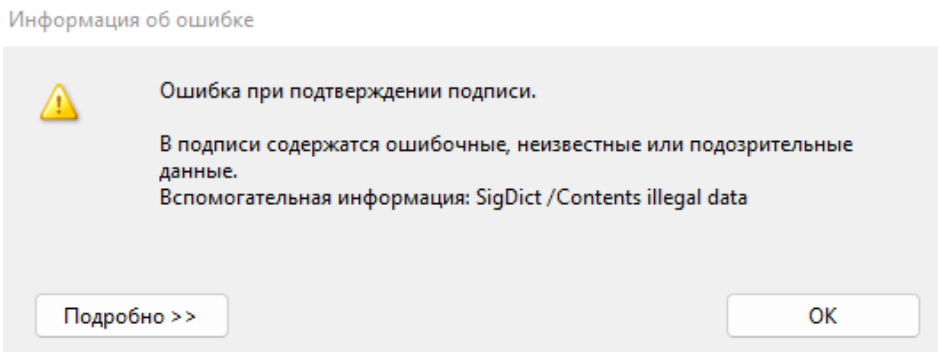


Рисунок 28 – Подтверждение подписи измененного pdf-файла.

Выводы.

В ходе выполнения данной лабораторной работы были исследованы алгоритмы создания и проверки цифровой подписи, а также алгоритмы генерации ключевых пар для алгоритмов цифрой подписи RSA, DSA, ECDSA.

1. Алгоритмы генерации ключевых пар:
    a. Была рассмотрена работа алгоритма RSA для генерации ключевых пар. Было определено, что в качестве открытого ключа алгоритм генерирует пару значений (e, n), а в качестве закрытого – значение d. Для генерации пары ключей алгоритм использует два больших простых числа p и q ($n = p * q$), которые после окончанию генерации уничтожаются. В качестве e выбирается любое число, взаимно простое с $\varphi(n)$. А значение d вычисляется из уравнения $e * d = 1\ mod\ \varphi(n)$;
    b. Была рассмотрена работа алгоритма DSA для генерации ключевых пар. Было определено, что в качестве открытого ключа алгоритм генерирует набор значений (e_1, e_2, p, q), а в качестве закрытого – значение d. Для генерации пары ключей алгоритм использует число p, размер которого составляет 512-1024 бита и кратен 64, и число q, размер которого совпадает с размером хэша и которое удовлетворяет равенству $(p - 1) = 0\ mod\ q$. В качестве e_1 выбирается такое число, что $e_1^q = 1 \bmod p$. В качестве d выбирается любое число, меньшее q, а значение e_2 вычисляется как $e_1^q \bmod p$;
    c. Была рассмотрена работа алгоритма ECDSA для генерации ключевых пар. Было определено, что в качестве открытого ключа алгоритм генерирует набор значений (a, b, q, p, e_1, e_2), а в качестве закрытого – значение d. Для генерации пары ключей алгоритм использует эллиптическую кривую, ее параметры a, b и p выбираются случайно, при этом p – простое число. В качестве e_1

выбирается любая точка на кривой. Значение q равно порядку циклической подгруппы группы точек эллиптической кривой ($q \times e_1 = O$). В качестве d выбирается любое число, меньшее $q - 1$. Точка e_2 вычисляется как $d \times e_1$.

d. Было измерено время работы рассматриваемых алгоритмов генерации ключевых пар. Алгоритм EC-239 показал наименьшее время генерации (0.012 секунд), а DSA-2048 – наибольшее (2.899 секунд). Алгоритм RSA-2048 показал время 0.894 секунд.

2. Процесс создания и проверки цифровой подписи:
    a. Была рассмотрена обобщенная схема создания и проверки цифровой подписи. Было определено, что для создания подписи требуется вычислить дайджест данных и зашифровать его закрытым ключом владельца сертификата. После ее создания сертификат вместе с подписью добавляется к данным. Для проверки данных достаточно вычислить дайджест полученных данных и сравнить его верификатором, который вычисляется при помощи расшифровки подписи открытым ключом сертификата.
    b. Было исследовано время генерации цифровых подписей при помощи алгоритмов RSA, DSA, ECSP-DSA и ECSP-NR. Алгоритм RSA показал наибольшее время генерации (0.01 секунд), а остальные алгоритмы – наименьшее время (0.002 секунды).
    c. Было исследовано время проверки цифровых подписей при помощи алгоритмов RSA, DSA, ECSP-DSA и ECSP-NR. Алгоритм DSA показал наибольшее время проверки (0.004 секунды при сохранении целостности и 0.012 секунд при нарушении целостности), алгоритмы ECSP-DSA и ECSP-NR показали время 0.002 секунд, а алгоритм RSA – наименьшее время (0 секунд).
3. Создание и проверка цифровой подписи алгоритмом ECDSA, основанным на эллиптических кривых:

a. Был рассмотрен процесс создания цифровой подписи. Было определено, что в качестве цифровой подписи алгоритм ECDSA генерирует набор значений (M, S_1, S_2) при помощи известного открытого ключа (a, b, q, p, e_1, e_2) и закрытого ключа d, где M – это подписанные данные. Для вычисления S_1 и S_2 выбирается секретное случайное число r, лежащее в диапазоне $1 < r < q - 1$. Далее находится точка на кривой $P = (u, v) = r \times e_1$. Значение S_1 вычисляется как $u \bmod q$, а значение S_2 – как $(h(M) + d \times S_1) \times r^{-1} \bmod q$, где $h(M)$ – дайджест M.

b. Был рассмотрен процесс проверки цифровой подписи. Было определено, что для проверки цифровой подписи необходимо вычислить верификатор V и сравнить его со значением S_1 – если их значения совпадают, значит подпись прошла проверку. Значения (M, S_1, S_2) получаются из цифровой подписи. Для начала вычисляется точка $T = (x, y) = (h(M) \times S_2^{-1} \bmod q) \times e_1 + (S_2^{-1} \times S_1 \bmod q) \times e_2$. Далее значение верификатора V вычисляется как $x \bmod q$.

4. Процесс создания цифровой подписи в среде PKI:

a. Был рассмотрен процесс создания цифровой подписи в среде PKI с использованием демонстрационной утилиты. Было определено, что схема создания цифровой подписи в среде PKI совпадает с обобщенной схемой создания цифровых подписей, рассматриваемой ранее. Цифровая подпись содержит зашифрованный дайджест вместе с информацией о алгоритме создания цифровой подписи и данными.

b. Была рассмотрена структура сертификата. Было определено, что сертификат содержит в себе версию сертификата, серийный номер от издателя, идентификатор алгоритма подписи сертификата, имя издателя и имя субъекта, период действия сертификата, открытый ключ (параметры и алгоритм генерации),

уникальные идентификаторы издателя и субъекта, дополнительную информацию об использовании ключа, а также цифровую подпись сертификата.

5. Подписание своего отчета:
    a. Был рассмотрен способ подписи PDF-документа и проверки его подписи при помощи сертификата средствами Adobe Acrobat Reader. При сохранении целостности PDF-документа подпись успешно проходила проверку, а после изменения PDF-документа – проверка на целостность проходила неуспешно.

Были получены практические навыки работы с рассматриваемыми алгоритмами с использованием приложения CrypTool 1 и 2.