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

ОТЧЕТ

по лабораторной работе №8

«Изучение цифровой подписи»

Студент гр. 8382 _____ Мирончик П.Д.

Преподаватель _____ Племянников А.К.

Санкт-Петербург

2021

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.

1. ГЕНЕРАТОРОВ 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. Зафиксируйте 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

RSA

Генерация двух больших простых чисел p и q (p и q держаться в секрете).

1. Вычисление n = p * q

2. Выбор произвольного e (e <n), взаимно простого с $\varphi(n)$.

3. Вычисление $d: e * d = 1 \, mod \, \varphi(n)$.

4. Числа (e, n) – открытый ключ, d – закрытый ключ, p и q уничтожаются.

DSA

- Выбирается простое число p, длиной между 512 и 1024 битами. Число битов в p должно быть кратно 64
- Выбирается другое простое число q, которое имеет тот же самый размер, что и дайджест - 160 битов, такое , что
$$(p-1) = 0 \; mod \; q$$
- 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)
- Выбирается целое $d < q$ и вычисляется $e_2 = {e_1}^d \; mod \; p$
- Объявляется открытый ключ (e_1, e_2, p, q)
- Назначается закрытый ключ d

Рис. 1.1 – Алгоритм генерации DSA

ECDSA

- Выбирается эллиптическая кривая $E_p(a,b)$, p – простое
- Выбирается точка на кривой $\boldsymbol{e_1}$= (x_1,y_1)
- Для дальнейших вычислений выбирается другое простое число q - порядок циклической подгруппы группы точек эллиптической кривой : $q \times (x_1,y_1)$=O
- Выбирается целое число d , $1 < d < q-1$ и назначается закрытым ключом
- Вычисляется другая точку на кривой $\boldsymbol{e_2} = d \times \boldsymbol{e_1}$
- Объявляется открытый ключ $(a, b, p, q, \boldsymbol{e_1}, \boldsymbol{e_2})$

Рис. 1.2 – Алгоритм генерации ECDSA

1.3. Исследование алгоритмов генерации

Сгенерируем ключи с использованием вышеперечисленных алгоритмов (табл. 1.1).

Табл. 1.1 – Время генерации ключей

Алгоритм	**Время**
RSA-2048	3.123s
DSA-2048	4.168s

EC-239	0.016s

Сгенерированные ключи представлены далее.

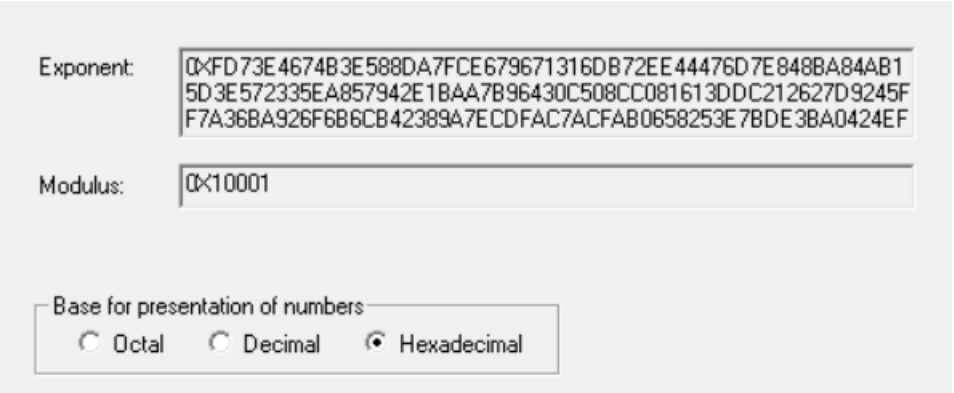

Рис. 1.3 – Ключ RSA

Открытые параметры ключа DSA:

```
DSA prime p (no. of bits = 2048):
  0  FFE2BB71 EFB1781C  F0380868 745C0473
 10  35B2403F EAD98C72  79650217 3B45CB4E
 20  047FD7DE E3FFC0F9  3F54A15A C9843ABC
 30  0E141850 9DB39B2C  75363277 2C222C12
 40  E1E4CFC2 299A6218  4AF178CB C2C57F66
 50  6408B3B0 D2D7C163  B289A6B4 B26FA57B
 60  73467185 FFDFD1A6  1C403E36 5FB1419A
 70  815A49DC 7646967C  DA34E458 265BD977
 80  3236A1D4 683F6708  2E695A78 267C14BC
 90  AA5F0123 0A33060C  0464C33F 19BF32C7
 A0  FAF39659 4A6FD4D8  79DE70FF B57470CC
 B0  32BC64C2 BB9EB48D  3D994583 BA0FEB34
 C0  94EBABD7 0B74A4B6  4F9BF376 091916A0
 D0  E07DAE94 5F52D69B  70E9D9A9 DEDC882D
 E0  F4107C79 241560D0  8340922B 6CAD2E12
 F0  B939B519 90BB067E  679B7F3E C0C76D3F
DSA prime q (no. of bits = 160):
  0  E746DFE0 2AEBDD38  1AA50FDE BB8C3384
 10  E5FB4FF5
DSA base g (no. of bits = 2048):
  0  D1586129 A8224AB9  9953DE5E 4C021446
 10  E0527B58 3ED3EB0F  5B4F85A5 1DC28FFB
 20  D28A3F86 BBE57A3F  B5EA0708 5F31FA34
 30  AB8D36FF 515F5B86  75F4630E B5305257
 40  5192D329 392C325C  0B973722 0A50F7A9
 50  5238DE7A 71EF2246  7F7DB48F 0CAD7310
 60  D15BA619 9093403E  72868482 3D752B21
 70  6BEA7152 E3B44CA9  B5CECEC0 474DE854
 80  4603F56A A7A3E12B  4AE9B645 A8B073B2
 90  93095721 E83AB1FE  3DA367F9 466ACB61
 A0  1B3DA53C E9D88509  E90EE2F3 83642474
```

```
 B0  7003B9B1 89069C79  D56145FE A110D2C1
 C0  EF38B5A6 BE43B76E  90B5F248 73F4592D
 D0  F415B284 669CFD15  917D62C4 9F774FB9
 E0  B68BAF54 2C0FA0C7  43F0BE8A F3832097
 F0  2E048E9A 188F828A  CE804356 3717B39F
Public y (no. of bits = 2048):
  0  8FDAFE5E 5B1568EF  F4D3E768 82F858E1
 10  4768D75E 1E8C57B5  12229D1D 17265C14
 20  377EBF24 000A7F0B  F3C937F6 7B28C384
 30  EE444103 4577CB2C  9E653D04 B6317ADB
 40  8025BA8A 4222AB26  80D7B0E6 1E5E93A6
 50  CCBF3CD4 44F26FEA  C761A0DD CA40F1A6
 60  9508EDD6 B1C9D4BD  CA8E0380 F747E9A1
 70  D4E97D13 ACBEBA80  0B7E8F60 06078A9E
 80  5F53F73C 6D3BDD41  C1F921B8 666931B4
 90  BF062689 648B245E  BDE5CD38 F0F1EC9E
 A0  6EB12619 F415F44F  BFCA63C2 5267450F
 B0  59DC21BF A939C0A3  7C877F8D DC19C67D
 C0  748F70D9 FE58DBD6  09A013A6 9ABE6137
 D0  066435CA BC4E77E5  2C648231 95CB1D79
 E0  3F90CE9D 57B3003F  DA5A55DA E3F88911
 F0  12973111 C486B99F  7134277F FF6B895B
```

Domain parameters of elliptic curve 'EC-prime239v3':

Parameters	Value of the parameter	Bit len...
	Elliptic curve E described through the curve equation: y^2 = x^3 + ax + b (mod p) :	
a	883423532389192164791648750360308885314476597252960362792450860609699836	
b	257710759664581349045614884019467140476383834471573891251575555059576126	238
p	883423532389192164791648750360308885314476597252960362792450860609699839	239
	Point G on curve E (described through its (x,y) coordinates):	
x	713702090966717781398151179513032310291275673609168278295934709018913114	239
y	152051417671827544218539107898347788360948341292488391005135896880966899	237
	G has the prime order r and the cofactor k (r*k is the number of points on E):	
k	1	1
r	883423532389192164791648750360308884771190369765922550517967171058034001	239

The public key W = (x,y) is a point on curve E and a multiple of G:	Bit len...
x = 415802376912681047269812820527211835503520043073652933657640560988183583	238
y = 849118957587297172359987717225744691689147770751328519112632680431999636	239

Рис. 1.4 – Параметры ключа EC-239

2. ПРОЦЕССЫ СОЗДАНИЯ И ПРОВЕРКИ ЦИФРОВОЙ ПОДПИСИ

2.1. 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.

2.2. Схема создания цифровой подписи

На рис. 2.1 представлена обобщенная схема создания цифровой подписи и ее проверки.

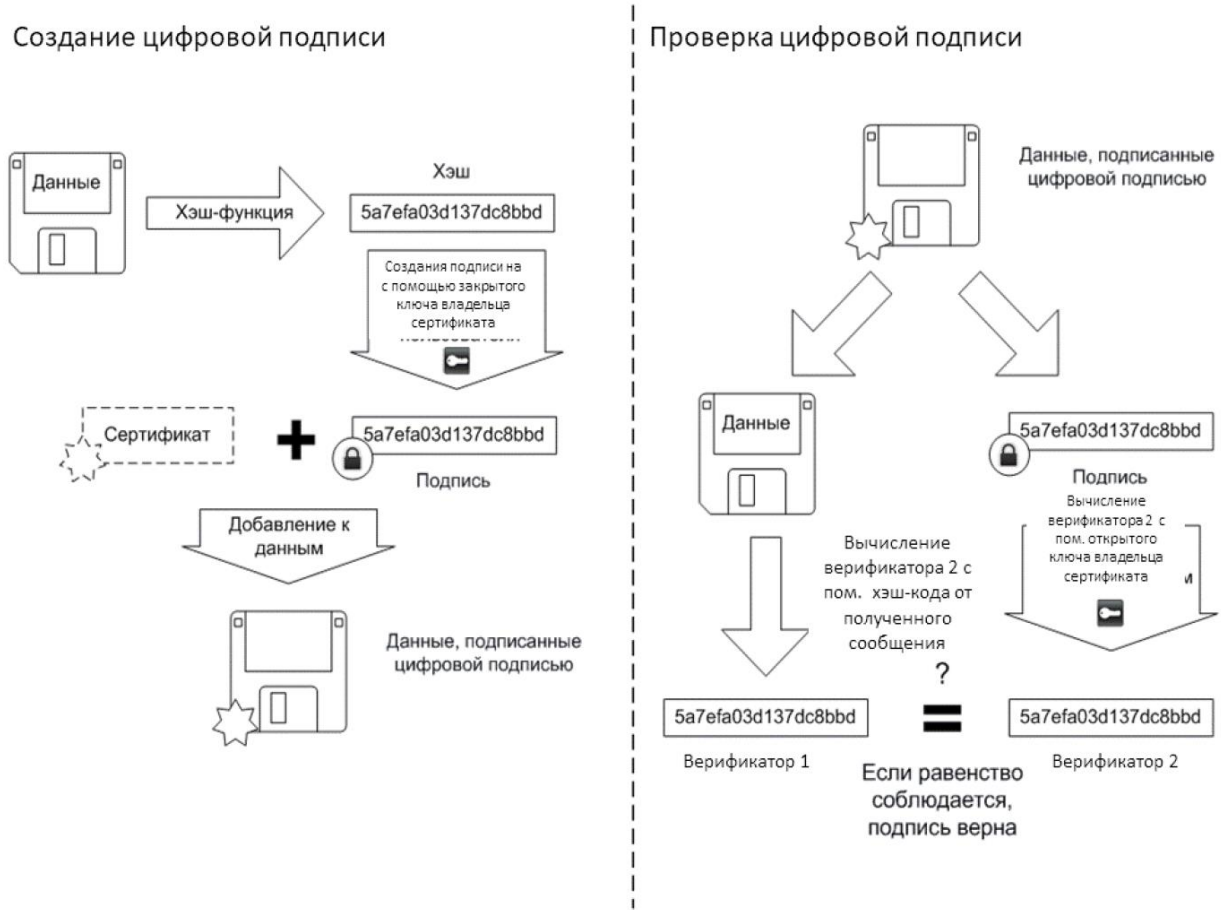

Рис. 2.1 – Создание цифровой подписи

2.3. Генерация подписи

Зафиксируем время, затраченное на подписание документа различными алгоритмами.

Алгоритм	Хеш-функция	Время
RSA-2048	MD5	0.012s
DSA-2048	SHA-1	0s
EC-239	SHA-1	0.006s

Сгенерированные подписи:

RSA:

```
8E258EC2A42FE617F4BFCDDA41EEEAB0309AD36F8047282A7E55EA6D7BF09E06
1EA925381BBB704A89A718C6C48B68289E2165D61E71F5E0EA33D15A7014B47C
A6E9651079B5C62B58F1CEC7527C2BF5B589051BF0487E94F871EE51B25C3800
169E584B4B4E41894F6853807B72CCD1928910C40C99B985885B963BF4823DA2
105EB1FFAA9706B21D88FFB1470C2BEF0F5F63C47D0F8432796409C24E0B2B8E
3FCE49CC3B8FC1510D988112E8CC2CF4D65ECCDE59F1310797FA2FE329C6E872
2C833CAE720419F11AD42C2CC32CAF137171D5C50FCCC05A093FE2DA52F717CD
66C95DF935F262A61C5135BF462EB9FB2130A2DA217AAD1D8D593458EBEEF7E1
```

DSA:

```
302C021445EBFC6C8AD0FE47AE038F093DD1F582567C0B6B02140E578BF571CE
D7EB1518AA06D6CB9E3BD2090200
```

EC-239:

```
c = 7FEDFEE5BF9119FE8C514FC3603F9268BF8785519732517473D7215B2B08
d = 5C64676FCFD3ED829784A8E7BEC73B2C38F916429D9BB33BC40EF6E99636
```

В результате проверки подписей было установлено, что все подписи валидны.

3. СХЕМЫ ЦИФРОВОЙ ПОДПИСИ НА ЭЛЛИПТИЧЕСКИХ КРИВЫХ

3.1. 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.*

3.2. Описание алгоритма

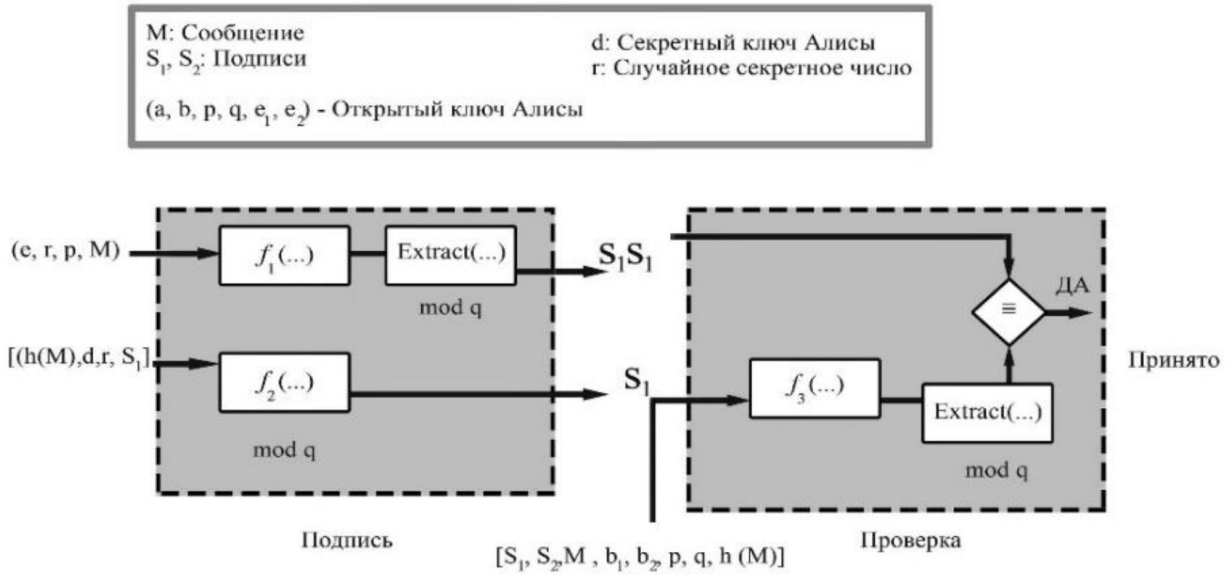

Рис. 2.2 - Схема цифровой подписи ECDSA

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)) (см. раздел 8.1), осуществляется подписание

документа, затем на принимающей стороне осуществляется проверка (рисунок 8.3).

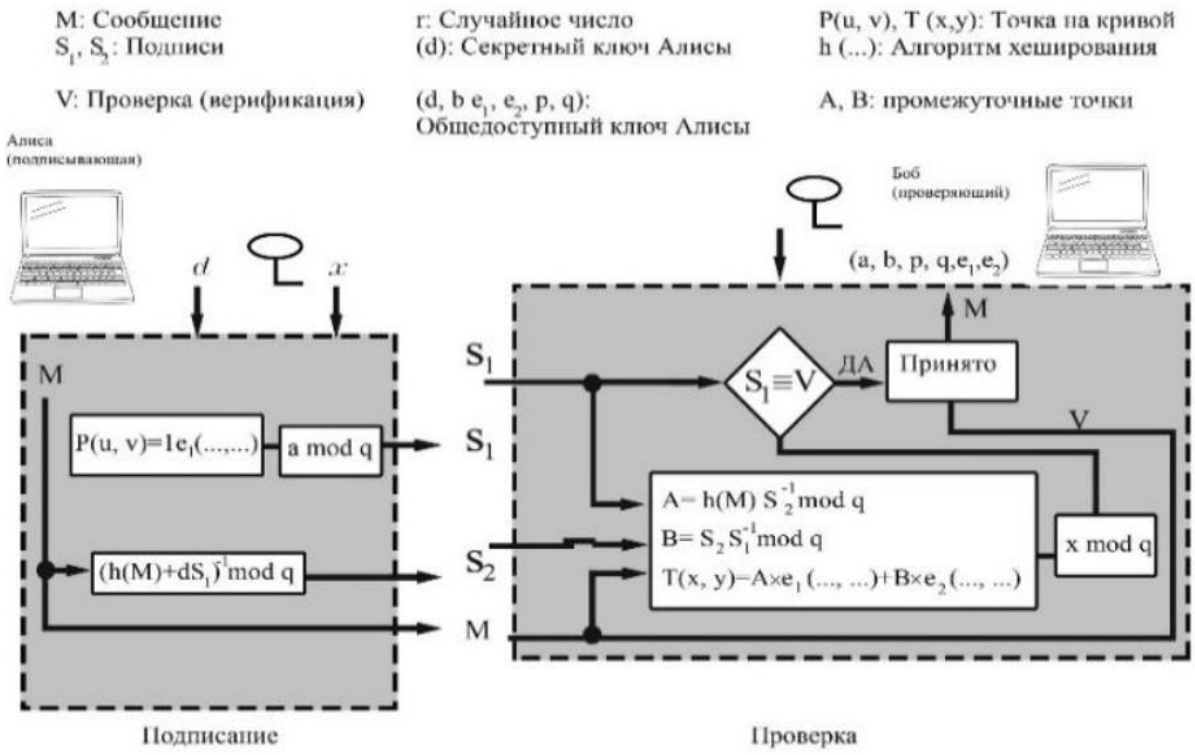

Рис. 3.3 – Подписание документа

Алгоритм подписания ECDSA состоит из следующих операций:

1. Выбирается секретное случайное число r: $r \in (1, q-1)$

2. Выбирается третья точка на кривой: $P(u, v)=r \times e1$

3. Вычисляется первая часть подписи по формуле:

$$S_1 = u \; mod \; q$$

где u- абсцисса.

4. Вычисляется вторая часть подписи по формуле:

$$S_2 = (h(M) + d * S_1) * r^{-1} \; mod \; q$$

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

Алгоритм проверки цифровой подписи ECDSA включает следующие операции:

1. Вычисляем промежуточные результаты A и B:

$$A = h(M) * s_2^{-1} \; mod \; q$$

$$B = S_2^{-1} * s_1 \; mod \; q$$

2. Восстанавливаем третью точку:

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

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

3.3. Подписание в CrypTool

Шаг 1:

```
Signature originator:  Pavel Mironchik

Domain parameters to be used 'EC-prime239v3':

  a  =
883423532389192164791648750360308885314476597252960362792450860609699836
  b  =
257710759664581349045614884019467140476383834471573891251575555059576126
  Gx =
713702090966717781398151179513032310291275673609168278295934709018913114
  Gy =
152051417671827544218539107898347788360948341292488391005135896880966899
  k  = 1
  r  =
883423532389192164791648750360308884771190369765922550517967171058034001

Secret key s of the signature originator:

  s  =
477740360151713420260984299698822165480014128220877492909819393467268 97

Chosen signature algorithm: ECSP-DSA with hash function SHA-1

Size of message M to be signed: 13103 bytes
```

Шаг 2:

```
Calculate a 'hash value' f (message representative) from message
M, using the chosen hash function SHA-1.

  f  = 145184490430019490027742136798169510169523357541 0
```

Шаг 3:

```
Create a random one-time key pair (secret key, public key) =
(u,V)
with the domain parameters of 'EC-prime239v3' (V=(Vx,Vy) is a
point on the elliptic curve):

  u  =
56951293364396066592382637097868852198277155490592128245214496780
5879787
  Vx  =
49988109520705430744034362920543975329924033539442528593127288181
3443224
  Vy  =
83143216845307161389080200360739414072145856623093418231510740919
9918255
```

Шаг 4:

```
Convert the group element Vx (x co-ordinates of point V on
elliptic curve) to the number i:

  i  =
49988109520705430744034362920543975329924033539442528593127288181
3443224
```

Шаг 5:

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

  c  =
49988109520705430744034362920543975329924033539442528593127288181
3443224
```

Шаг 6:

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

  d  =
74057015564757647123580457602396013925581314378428354426035496379
3712721
```

3.4. Проверка лекционного материала

Выберем кривую $E_{67}(2,3)$, $e_1 = (2, 22), d = 4$. Найдем $e_2 = d * e_1 = (13, 45)$ – рис. 3.4.

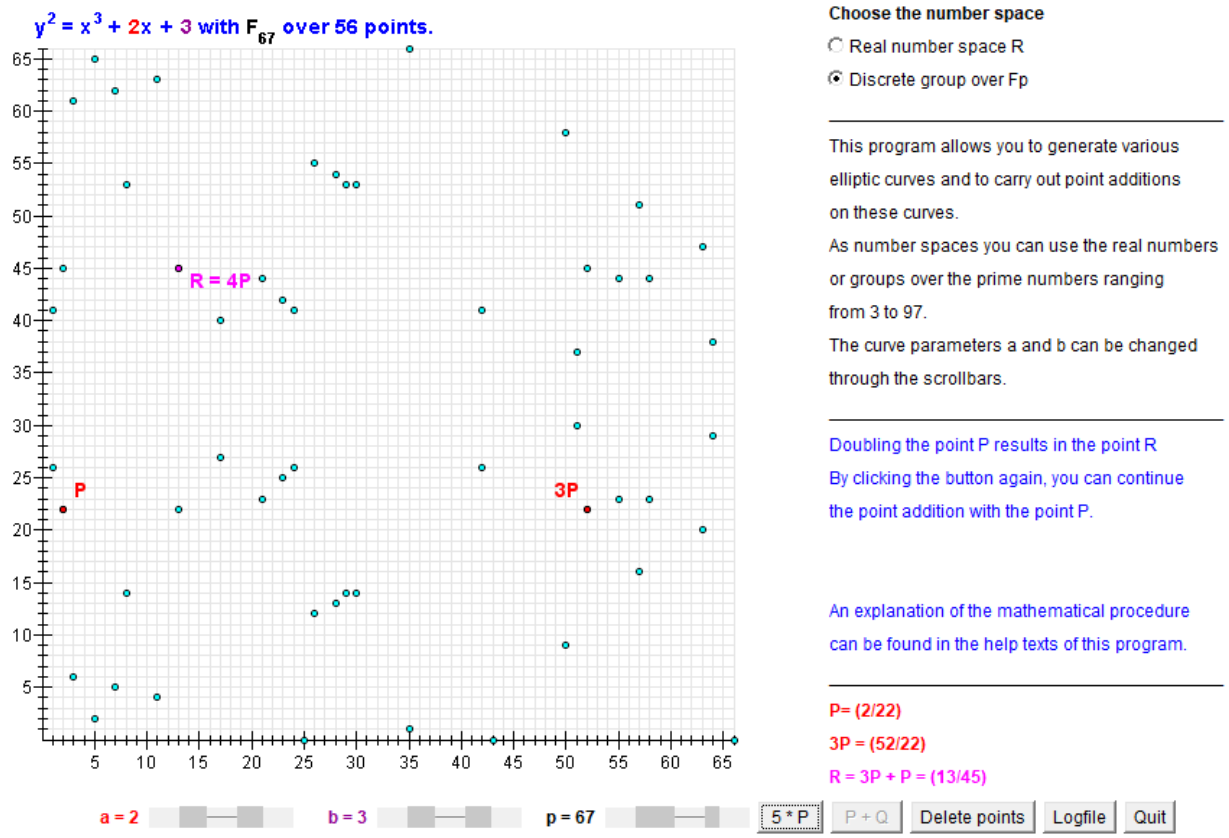

Рис. 3.4 – Поиск e_2

Пусть текст представляет собой точку $(24, 26)$. Выберем случайное $r = 2$. Найдем $C_1 = r * e_1 = (35, 1), C_2 = P + r * e_2 = (24, 26) + (23, 25) = (21, 44)$:

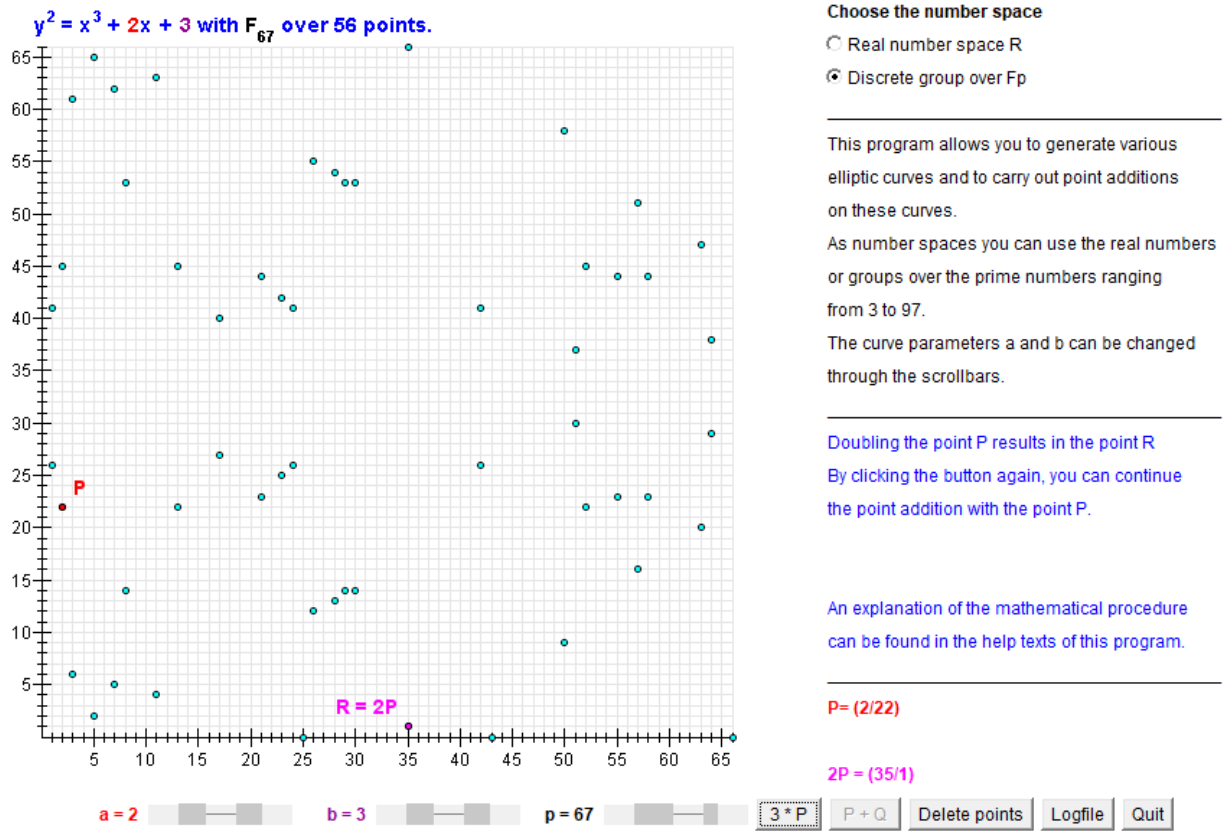

Рис. 3.5 – Поиск C_1

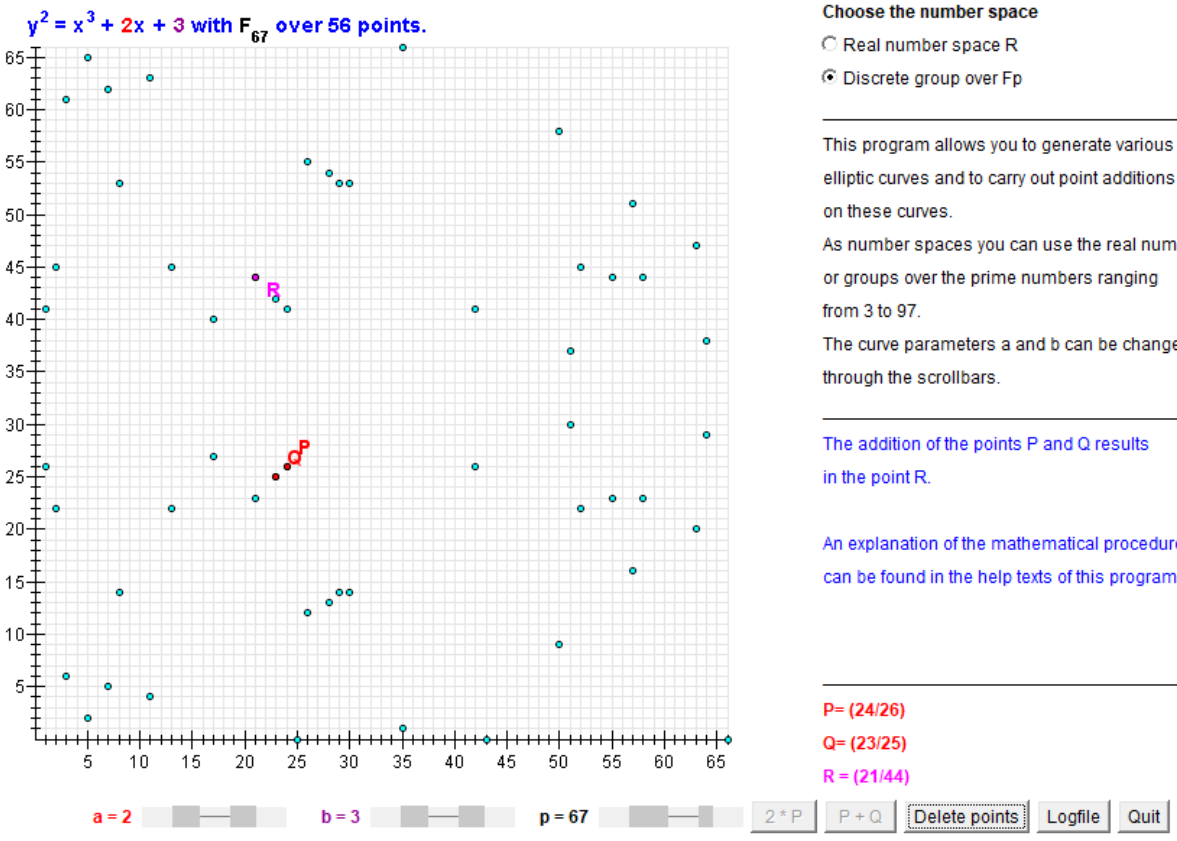

Рис. 3.6 – Поиск C_2

И расшифруем полученный шифротекст:

$$P = C_2 - d * C_1$$

$$d * C_1 = (23, 25) - \text{рис. } 3.7$$

$$-(23, 25) = (23, 42)$$

$$P = (24, 26) - \text{рис. } 3.8$$

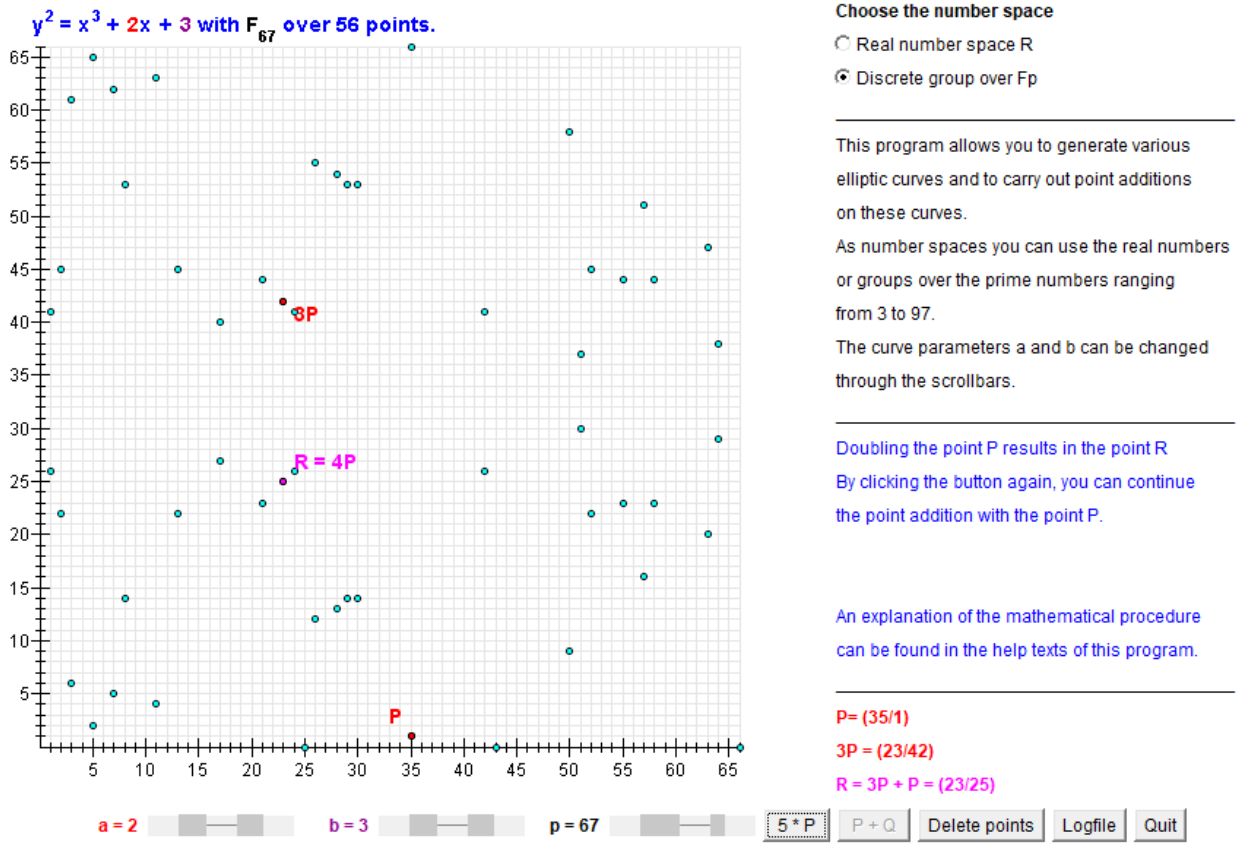

Рис. 3.7 – Поиск $d * C_1$

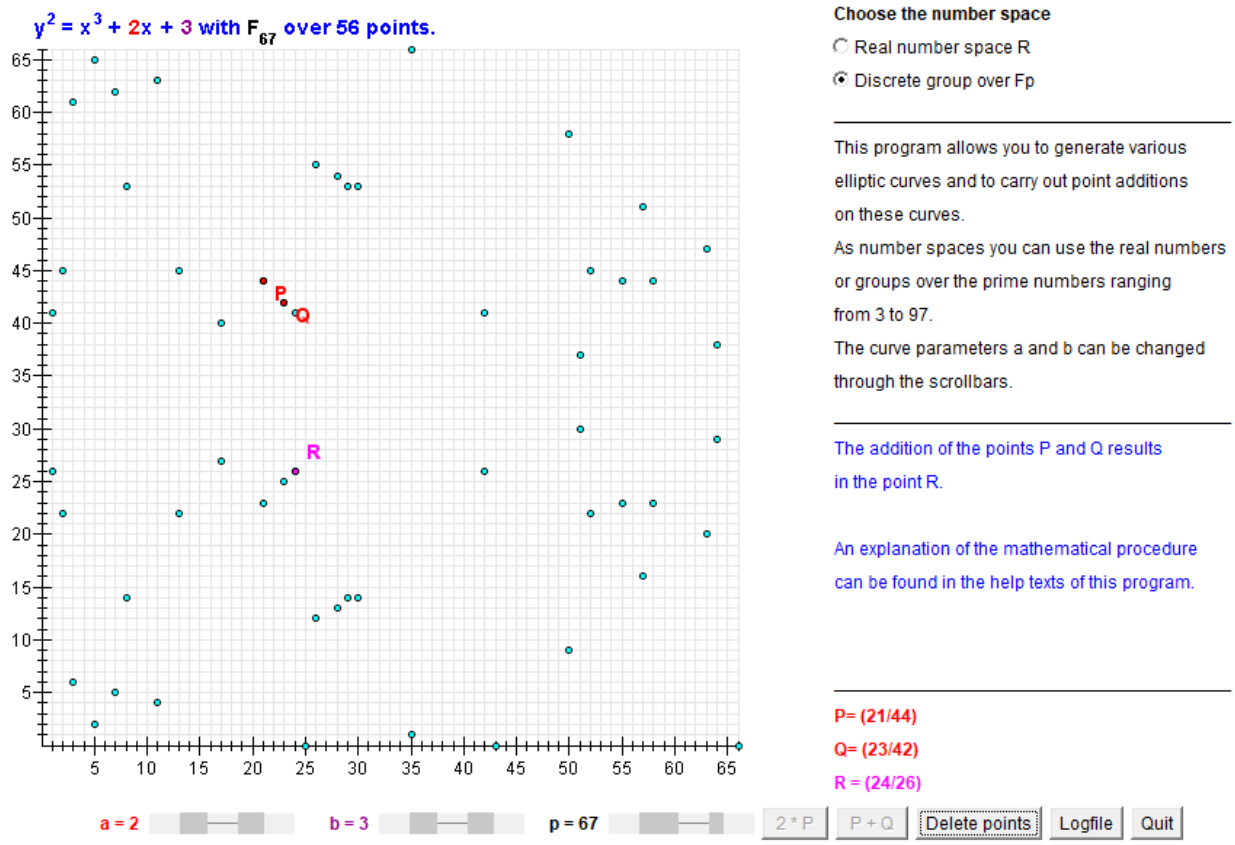

Рис. 3.8 – Поиск P

Лекционные вычисления сошлись с проведенными в программе CrypTool.

4. ДЕМОНСТРАЦИЯ ПРОЦЕССА ПОДПИСИ В СРЕДЕ PKI

4.1. 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.

4.2. Процесс подписания документа

CrypTool позволяет ознакомиться с процессом подписания поэтапно: выбор хеш-функции, выбор ключа, подписание (зашифрование хеша), добавление сертификата и т.д. Интерфейс программы представлен на рис. 4.1.

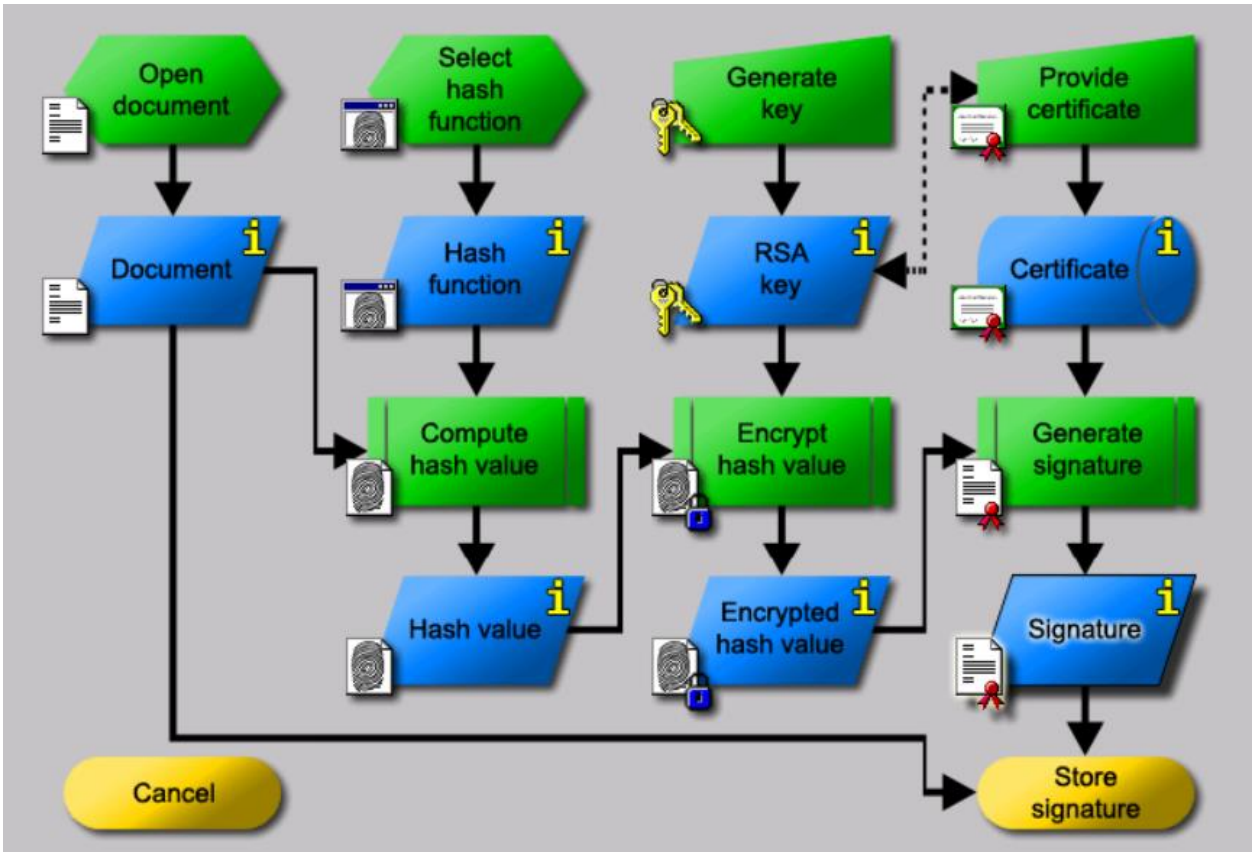

Рис. 4.1 – Визуализация процесса подписания сообщения

4.3. Сертификат

```
Version:                    2 (X.509v3-1996)
SubjectName:                CN=Pavel Mironchik [1639985057],
DC=cryptool, DC=org
IssuerName:                 CN=CrypTool CA 2, DC=cryptool, DC=org
SerialNumber:               6A:1E:D3:4F:A0:2E:C0:63
Validity  -  NotBefore:     Mon Dec 20 10:24:20 2021
(211220072420Z)
              NotAfter:     Tue Dec 20 10:24:20 2022
(221220072420Z)
Public Key Fingerprint:     76A5 A755 7D9A 612D 628E 943C E999
5BAD
SubjectKey:                 Algorithm rsa (OID 2.5.8.1.1), Keysize
= 2048
              Public modulus (no. of bits = 2048):
```

```
                 0  FD73E467 4B3E588D  A7FCE679 671316DB
                10  72EE4447 6D7E848B  A84AB15D 3E572335
                20  EA857942 E1BAA7B9  6430C508 CC081613
                30  DDC21262 7D9245FF  7A36BA92 6F6B6CB4
                40  2389A7EC DFAC7ACF  AB065825 3E7BDE3B
                50  A0424EF0 9399428A  05F3F742 82638ED6
                60  7404A442 9013D980  87638709 57F83564
                70  419630FB A01F382C  A80EEF6E AAAAA42A
                80  8B65D76A D0857522  B0241C98 9B460113
                90  AFA95602 22B9F9BE  506F842C 16B175FE
                A0  AAAEF757 65CCED13  90F2E77B 687096D9
                B0  76030C0F C3A89687  C4CDF617 1E31246E
                C0  532F1731 55A454D5  9CA7ECE6 650F5813
                D0  EF7457EC 26A9982A  6B6DF9BB FA62EF26
                E0  A3BBE913 08BAFB39  08431575 6D56C587
                F0  C30EA8D4 FF783EFA  3FF3E88C 2A03DE43
              Public exponent  (no. of bits = 17):
                 0  010001
Certificate extensions:
Private extensions:
    OID 2.206.5.4.3.2:
        PrintableString:
                |[Mironchik][Pavel][RSA-2048][163|
                |9985057][RSA]                   |

SHA1 digest of DER code of ToBeSigned:
                 0  0139B998 1E13C026  93C0DE8C 52AF8939
                10  0692275C
Signature:               Algorithm sha1WithRSASignature (OID
1.3.14.3.2.29), NULL
                 0  2350AEEF 5DB45522  0480E4F0 2E82DB27
                10  58ABB97A E51C53FF  406891EB A3F03E1C
                20  415C7C41 5E1731B3  832D2701 402EEE0C
                30  4CF722DC 6C25E61B  6241DE28 BA819AA0
                40  2B8447CE AA969E50  E28E9946 082ABB39
                50  B5E4B50F 52D325F1  8FCA6A3E C2F62804
                60  37E6A16E 53195066  7328C7A0 1EE2A350
                70  7E37FB98 84226EDA  EC09913B 57906A5D
                80  99DD4D9B EF479E33  AF039500 6108FB86
                90  B311241D 15BFE311  D6376495 1B118F00
                A0  FC66DE72 317D3EA9  30DB4EBD 202D51D8
                B0  BA29AC13 6839A0BC  4278DB4A 47C1B5F0
                C0  B2EE2462 EC1CC4B7  C3BF4082 CC4ACBDF
                D0  9045B45B F9920611  D122F45A 1103E23D
                E0  2988597C 395DAC3A  C61DE124 E8DE290B
                F0  419259AB 9E3042AD  5184A9C0 E9F9C34D
Certificate Fingerprint (MD5):
BC:73:5A:87:F2:F7:09:8F:1C:87:96:1B:1D:B5:53:D5
Certificate Fingerprint (SHA-1):  9596 A3D3 EB5F D584 356F BE0E
C525 49AF 0AC8 75B9
```

Для используемого ключа используется сертификат, записанный в определенном формате.

Стоит отметить некоторые отличия сгенерированного сертификата от представленной в лекции структуры: в основном в первых пунктах. Отличается порядок пунктов, например, идентификатор алгоритма подписи находится непосредственно перед информацией об открытом ключе.

5. ПОДПИСАНИЕ СВОЕГО ОТЧЕТА

5.1. Задание

1. Сконвертируйте отчет в формат pdf.

2. Экспортируйте ранее созданный сертификат ключевой пары *RSA Digital Signatures/PKI->PKI/Generate…->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.

5.2. Подписание отчета CrypTool сертификатом

В процессе подписания отчета сертификатом, сгенерированным в CrypTool, возникла ошибка, указывающая на невозможность использования сертификата по причине неподдерживаемого алгоритма открытых ключей (рис. 5.1).

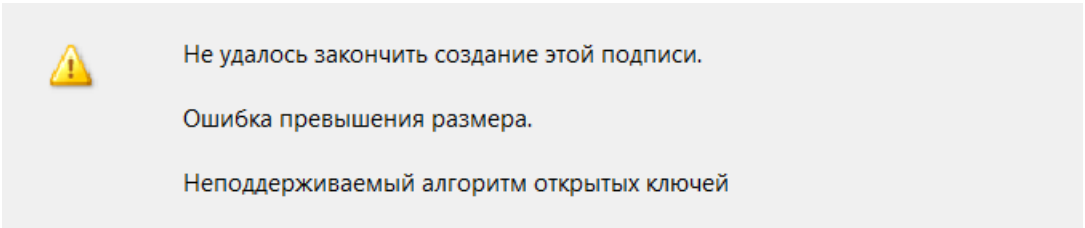

Рис. 5.1 – Ошибка подписания

5.3. Подписание сгенерированным в Adobe Reader сертификатом

Был сгенерирован сертификат в среде Adobe Reader (рис. 5.2). С его помощью был подписан pdf документ.

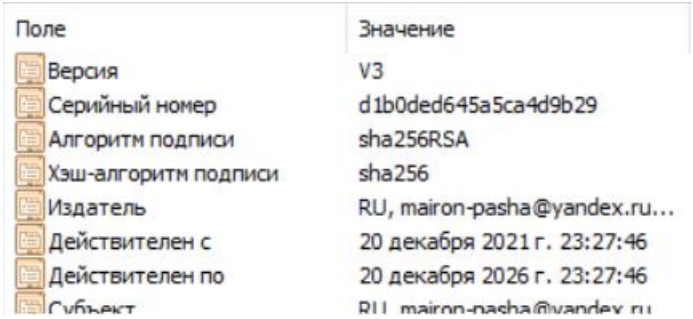

Рис. 5.2 – Свойства сертификата

Далее выполнена попытка изменения документа внесением в него дополнительного комментария. После добавления изменения и проверки подписи появилось сообщение, что в документ были внесены изменения с момента подписания (рис. 5.3).

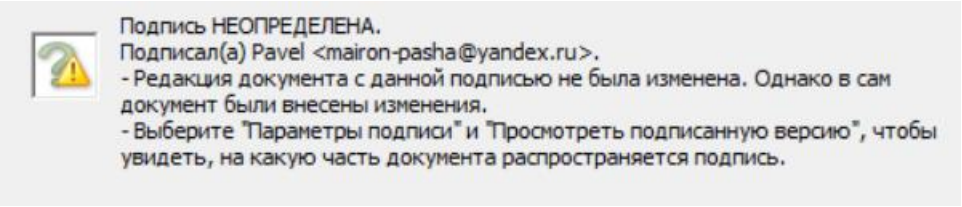

Рис. 5.3 – Проверка измененного документа

ВЫВОДЫ

1. Рассмотрены генераторы ключевых пар для алгоритмов RSA, DSA и ECDSA. Определено, что RSA и DSA выполняют генерацию за примерно одинаковое время (3-4 секунды), в то время как ECDSA значительно быстрее – за 16мс.

2. Рассмотрен общий алгоритм создания цифровой подписи документа: по сообщению генерируется хеш, затем он шифруется закрытым ключем, а для проверки подписи необходимо расшифровать хеш открытым ключем и сравнить его с хешем сообщения, по которому выполняется проверка. Экспериментально выяснено, что время создания подписи для документа достаточно невелико – подпись документа длиной 5000 символов заняла от 0мс (DSA) до 12мс (RSA).

3. Рассмотрен алгоритм цифровой подписи на эллиптических кривых. Проведено подписание документа и проверка соответствия документа его подписи, а также проверка того, что измененный документ подписи не соответствует. Повторены лекционные шаги по зашифрованию и расшифрованию сообщения с использованием средств визуализации CrypTool и подтверждена корректность вычислений.

4. Выполнено пошаговое создание подписи документа с использованием подпрограммы CrypTool - пошаговой визуализации подписи документа. В результате был сгенерирован сертификат. После сравнения его структуры с представленной в лекции оказалось, порядок пунктов отличается.

5. Проведена попытка подписания документа с использованием экспортированного из CrypTool сертификата. Подписать документ не удалось в связи с “неподдерживаемым алгоритмом открытых ключей”. После этого был сгенерирован сертификат средствами Adobe Reader, которым был успешно подписан документ. Затем в подписанный документ были внесены изменения, что подтвердилось при проверке подписи.