Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по лабораторной работе №3.2
по дисциплине «Защита информации»

Студент Лукьяненко В.А.

Группа ИУ7-71Б

Преподаватель Руденкова Ю.С.

2025 г.

1 Задание

1.1 Цель работы

Цель работы: создание электронной цифровой подписи. Шифрование и расшифровка произвольного файла.

1.2 Содержание работы

Для выполнения данной лабораторной работы необходимо решить следующие задачи:

1. реализовать программу создания и проверки электронной подписи для документа;
2. обеспечить шифрование и расшифровку произвольного файла с использованием разработанной программы;
3. предусмотреть работу программы с пустым и однобайтовым файлом;
4. протестировать работу программы на различных типах данных.

2 Теоретическая часть

Вопросы для защиты работы

1. Дайте определение электронной цифровой подписи.

Электронная цифровая подпись — это реквизит электронного документа, получаемый в результате криптографического преобразования информации с использованием закрытого ключа. ЭЦП обеспечивает:

— аутентификацию автора документа (подпись может быть создана только владельцем закрытого ключа);

— целостность данных (любое изменение документа делает подпись недействительной);

— невозможность отказа от авторства.

2. Опишите алгоритм создания цифровой подписи.

Algorithm 1 Алгоритм создания цифровой подписи

Require: Документ M, закрытый ключ K_{priv}
Ensure: Подпись S
1: Вычислить хэш документа: $h = H(M)$.
2: Зашифровать хэш с помощью закрытого ключа: $S = h^d \pmod{n}$.
3: Сохранить подпись S вместе с документом.

3. Опишите алгоритм проверки цифровой подписи.

Algorithm 2 Алгоритм проверки цифровой подписи

Require: Документ M, подпись S, открытый ключ K_{pub}
Ensure: Результат проверки (действительна или нет)
1: Вычислить хэш документа: $h = H(M)$.
2: Расшифровать подпись: $h' = S^e \pmod{n}$.
3: Сравнить h и h':
4: **if** $h = h'$ **then**
5: Подпись корректна.
6: **else**
7: Подпись недействительна.
8: **end if**

3 Практическая часть.

Листинг 3.1 – Файл `main.py`,

```python
from Crypto.PublicKey import RSA
from Crypto.Signature import pkcs1_15
from Crypto.Hash import SHA256
import os

def generate_keys():
    key = RSA.generate(2048)
    private_key = key.export_key()
    public_key = key.publickey().export_key()

    with open("private.pem", "wb") as f:
        f.write(private_key)
    with open("public.pem", "wb") as f:
        f.write(public_key)

def sign_file(input_file, signature_file, private_key_file):
    with open(private_key_file, "rb") as f:
        private_key = RSA.import_key(f.read())

    with open(input_file, "rb") as f:
        data = f.read()

    h = SHA256.new(data)

    signature = pkcs1_15.new(private_key).sign(h)

    with open(signature_file, "wb") as f:
        f.write(signature)

def verify_signature(input_file, signature_file, public_key_file):
    with open(public_key_file, "rb") as f:
        public_key = RSA.import_key(f.read())

    with open(input_file, "rb") as f:
        data = f.read()

    with open(signature_file, "rb") as f:
        signature = f.read()
```

```
39
40      h = SHA256.new(data)
41
42      try:
43          pkcs1_15.new(public_key).verify(h, signature)
44          print("Подпись␣корректна")
45          return True
46      except (ValueError, TypeError):
47          print("Подпись␣недействительна")
48          return False
49
50
51  if __name__ == "__main__":
52      if not os.path.exists("private.pem") or not
            os.path.exists("public.pem"):
53          generate_keys()
54      sign_file("input_eng.txt", "signature_eng.bin", "private.pem")
55      verify_signature("input_eng.txt", "signature_rus.bin",
            "public.pem")
```

4 Пример работы программы

Для начала создаются ключи:

lab3.2 > src > public.pem

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzrdrZ9KD2b1OQcuR9nc0
HlpLf+dCd0xkRTHkSsKKTjzP7IoP3dxVNTkPmEeJBxXK7pwxRscySdwTY05qVRtX
MP/rEreyplteQzBiVnizmNjgjKX0EfeiYhi8T8rNou4Hj2ouOQLURN7cHPxDPaW3
9jGR9QZwSIW0884Kp+YA4ZfLvwbDihwR4UvwpYui0MfS7xQat+u8HkyfXsP60SXE
ODp4vS2ThMYTn9UNAu1f9Q8Mu37WN4nA/PQESDz2iK17+CP7bIpiqwxZKXKGmZwi
S5QDIkosW5pO5o48kphN8cRsKDyhmo79hJTpbiG3ly4dwtgcsm0wQS9rWLgrVTSv
iQIDAQAB
-----END PUBLIC KEY-----
```

Рисунок 4.1 – Публичный ключ

lab3.2 > src > private.pem

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAzrdrZ9KD2b1OQcuR9nc0HlpLf+dCd0xkRTHkSsKKTjzP7IoP
3dxVNTkPmEeJBxXK7pwxRscySdwTY05qVRtXMP/rEreyplteQzBiVnizmNjgjKX0
EfeiYhi8T8rNou4Hj2ouOQLURN7cHPxDPaW39jGR9QZwSIW0884Kp+YA4ZfLvwbD
ihwR4UvwpYui0MfS7xQat+u8HkyfXsP60SXEODp4vS2ThMYTn9UNAu1f9Q8Mu37W
N4nA/PQESDz2iK17+CP7bIpiqwxZKXKGmZwiS5QDIkosW5pO5o48kphN8cRsKDyh
mo79hJTpbiG3ly4dwtgcsm0wQS9rWLgrVTSviQIDAQABAoIBACj23ay5IcJWOqoc
uIvuQvCJ5cDOuBPymkNBGBO9mhFXiT+/e1X0uhj+OgosEOpLxgeDgdtcEanj2B67
lSRQ8vYlnPS8XQWNATDQ54fpyvNs9RDCOUwPNgl5C+ZHEvmo8GwZ91d6aUzNmjKP
RhbzHnHGEH3NPRaOEpYm/GMZ2A5cPiwCRLHafK4Avbn5v3dA6xX7GI8G41SoWYVX
s9jskh1T3Vo2LIWC2uOu1cvL7Suk85pSuEnVe9UAoe49qBBaSYj9xu3gJL1Zw4WF
Lg+WfE40T4P+WmNHGCdQC/z+tdX1ngZUCyUe3/xFv3pgJHMV8CC4C+8TqiUx9QQf
iSjUy/UCgYEA0ksRmVJnQ1OtQ8y+v+f0bIryBO/jV0t/x2XeZuL/rRfYlQF05NOa
dRWof/9AUOYq4L86uwJMsWi5viS6Tl/kd8cDmDZUSrr+F7xDGaSc6rHTu95udAyN
tUYZE8TLyKDKnW2bO6lLQ9TcGoj7mM+FSgyukTa2l381Iyp9c//XnU0CgYEA+6VW
Jagpm4ZfV5MxsV6Brb1KTQzmwPl2A+kfgbHLGcPSZSiNxT5T7724cJjxZRGKRF+H
CjXl/wbEiue/dhYuLH3lrtPZSfD7aKlvUb1k+IUEX3H8cZDpPSB3TVGi5KNx/+QT
6JkKL1omKdZWkek+0s0/AL2NmhsWSHmEVqaFrS0CgYATSOMSRmZF3FSVVR9hBep6
UbukDEOy66+KH6NnmAWOYBNyhrGrkmQlvJi5NDU3tgPUnHTMVYem9U2jTEbxjQWs
nkmMGd9dAwzuLEuS/G+PbkNWhS1n4wbTytvd3jw559Ts3vJPZFUMoWaV4F4CQ3Wb
ttIDaxsPhZDf6rOaganKOQKBgEro4+Uf0wr2DCmwqolHPckvYQ/guExXtlbhrbxY
5cHMtjPgx5sgu7HRQzdZf/p7JzLFURisBMRPCnDjHe6UGbiq3ZWWgMdvEhgM4BNw
Er+cMToMX36bCaNlp73/gEMZmJT5GygRbqO77UJlCK40xbvQ12FsC9wFFjQK4Gc+
+OodAoGBALwqGKB6SweY8qPlq8yP/Npy2dsoMxktEUCaNocK5hAm77Usae6OXCt2
JSoT0WT6gZH7kncs08o8kc2Ivp3hUonZ43p5nvzoSeWwFybIj5f7QvvnePar0IeF
/FCdOYnjdnr8rDuZQMcUMikpEWSIbvHCeidDpXa8eFKwjugb5Rwl
-----END RSA PRIVATE KEY-----
```

Рисунок 4.2 – Приватный ключ

Также есть два текстовых файла на 10000 символов:

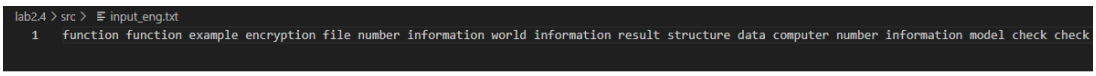

Рисунок 4.3 – Английский файл

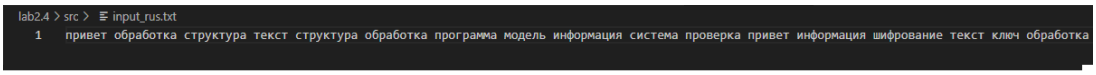

Рисунок 4.4 – Русский файл

В результате получается два файла электронной подписи:

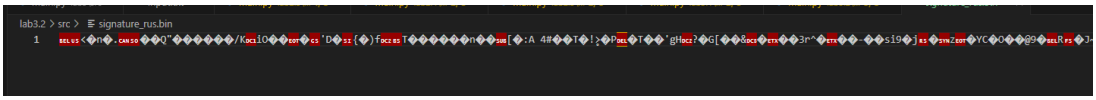

Рисунок 4.5 – Подпись для английского

Рисунок 4.6 – Подпись для русского

Если проверить подпись для соответствующих файлов, получим вывод:

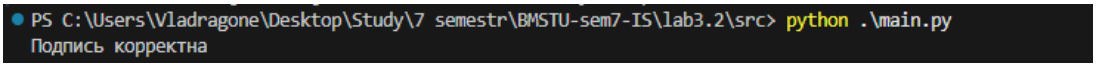

Рисунок 4.7 – Совпадение подписи

А если попробуем сравнить английскую подпись для русского файла или наоборот:

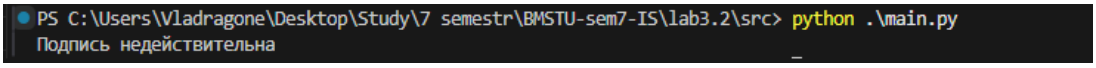

Рисунок 4.8 – Не совпадение подписи