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 №3.3
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 подписи. Шифрование и расшифровка архивных файлов.

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 архивного файла (rar, zip или др.) с использованием разработанной программы;
3. предусмотреть работу программы с пустым и однобайтовым архивом;
4. протестировать работу программы на различных архивах.

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. Дайте определение электронного сертификата.

Электронный сертификат — это электронный документ, удостоверяющий соответствие открытого ключа конкретному владельцу. Сертификат содержит:

— данные о владельце (ФИО или название организации);

— открытый ключ пользователя;

— информацию об органе сертификации (CA);

— срок действия сертификата;

— цифровую подпись удостоверяющего центра.

3. Какие существуют модели организации инфраструктуры электронных сертификатов?

Выделяют следующие модели организации инфраструктуры открытых ключей:

— **Централизованная модель** — используется единый удостоверяющий центр (CA), который выпускает сертификаты для всех пользователей.

— **Иерархическая модель** — корневой центр сертификации делегирует полномочия нижестоящим центрам, образуя древовидную структуру доверия.

— **Сетевая модель** — доверие основывается на перекрёстной сертификации между несколькими центрами, образующими сеть доверия.

— **Web of Trust** — децентрализованная модель, где пользователи самостоятельно подтверждают ключи друг друга.

3 Практическая часть.

Листинг 3.1 – Файл `main.py`,

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

4 Пример работы программы

Для начала создаются ключи:

lab3.3 > src > public.pem

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAihYwBffEaxeh1ap+qgmV
5GvUJnZmYA2f71ywOW0VRWBlQJJIOrg1I53YzndA3y77lMyc79lbVu96DDms50Wo
/7UVYZhtq3xJKJ67R7T8ILJcn01lMbFx1xEuYgF+mcemarXI2CB208FPJFOUZm5q
HdVA9syIUiKwfZJReGJbmmGkz7EXL0LhB4IFLY+ZI6EhV7xxNbjZBULfEHPN1BD6
HcFZ1m8qKpPEPSxQj2ucZ5ASOGVFrJ238NZzVfcTzceM64REB28cqpxzF2qjQ/am
xlHYmO9b7aCqMd/AVGruEyAIWlmBGE1CD/TjEnBOqxeg0U/RqBGtvH6t1g+JJ2c/
hQIDAQAB
-----END PUBLIC KEY-----
```

Рисунок 4.1 – Публичный ключ

lab3.3 > src > private.pem

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAihYwBffEaxeh1ap+qgmV5GvUJnZmYA2f71ywOW0VRWBlQJJI
Org1I53YzndA3y77lMyc79lbVu96DDms50Wo/7UVYZhtq3xJKJ67R7T8ILJcn01l
MbFx1xEuYgF+mcemarXI2CB208FPJFOUZm5qHdVA9syIUiKwfZJReGJbmmGkz7EX
L0LhB4IFLY+ZI6EhV7xxNbjZBULfEHPN1BD6HcFZ1m8qKpPEPSxQj2ucZ5ASOGVF
rJ238NZzVfcTzceM64REB28cqpxzF2qjQ/amxlHYmO9b7aCqMd/AVGruEyAIWlmB
GE1CD/TjEnBOqxeg0U/RqBGtvH6t1g+JJ2c/hQIDAQABAoIBAAGunYJ6by5tc5o6
YR/0d+/4DvXc/AhFgjeUkHen8toNKjlBeZca3ALaubkMzix/J69mAq9lSBiPWD7t
NkRRnpURgU4sC4H1mfU1tyBQ0xVu4XP1/w/o58hv5s8TfOr7T0qo0OaDwWQjobGB
Qc3cSyH2kWZa6CTf/qMOSjxw2PQmIcVQIIRNU2tTi8jwFPAPihA361UgRpAz/IiD
/X3oCxLMRvhq6buBC+1VEKMvPN4BtmXXj42PikMturMPt5jNs8OBzSYO+YVdINBg
e9jV0yOxDQDIDF29XclhrQHBILiWVv2o7d2mbygUGvXHOklVKn3y4lYgYa2/MFbK
pAHn2y0CgYEAuJwrdIMlrnZyAhzRILs3u8VsW+hy7bSYmYIS01U06Q14OoeOgywe
exfaoQoIFxXM3oc0oWBbUZrni9El9aymThgG75zl+qzd9njxuno4DfS1BCPKvFvV
wpsdbxxiWOPQjYpUt6KScSJpkcaEqTjGIen4wgyutEbyV+sghuMUYu8CgYEAv3xY
sImISHvNfbnivui9XzxhHZ/q15nQn2VhFUBDwXtiGg++Su91JcA0TP72Q2Wr++6O
yqHZEEXnCJ1f7K+7Mq43v5BxHFMMI2sR5zB2qyUjm8DN44MgrMYR00WsE/CnXwV9
MId765e8J1OOawMdfzAC2PutbpGYflQDKyMf9MsCgYEAlxmz86x4RiDKVfJSNVEn
X/hKJbvPfcIoeNTfJbyDDxGoS3eybKjK1gODPfYhAwwj4CuYJ1HfzJ5WUllCMaA7
ioXd/3dTQWcGUftSF0h74uhc1OOax6r+/kbJw6zezlTA2WjUyG03IYSCFf//iyZd
/Vos8l9BC2taZh2cROtOP18CgYBSvgmY5yUTpSOLg2pecjV6VJ3mbjbNSTLj3avQ
wf9FOOqQNmNVABoP4Pj4mGeTgfiKGXBsUPl51arahx5vKRRr9uoyBdlKkvw4xchr
rP5qP9ceM3cEK6JP5hrHv2B3EVNRdQ6+jJzYla2BFNMCnMr/LaDdYAt2LsXFOMpR
tpDVdwKBgBcE6pYGrjptqoXS4DxavOOxth366F2DUHqAVn+elgBTiqlOxBq05zpQ
OjsHJ0lLoVmm39EE6Aepvz6RMD0IDIpcNB7QMnS5+B15bmbBubG5AVwUBuPHG0qr
wchVaic/V8hNZuMpkd0zFrJlZFLZYvil4VbyXbHAzzFBCdP/ywyi
-----END RSA PRIVATE KEY-----
```

Рисунок 4.2 – Приватный ключ

Также есть архив:

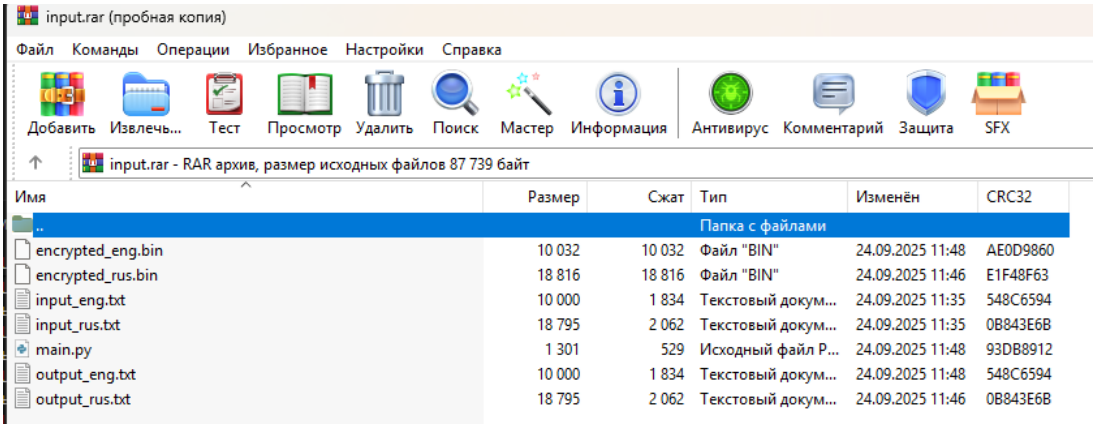

Рисунок 4.3 – Входной архив

В результате получается файл электронной подписи:

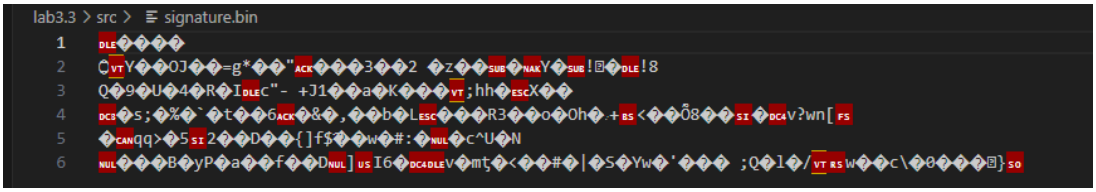

Рисунок 4.4 – Подпись для архива

Если проверить подпись для соответствующего архива, получим вывод:

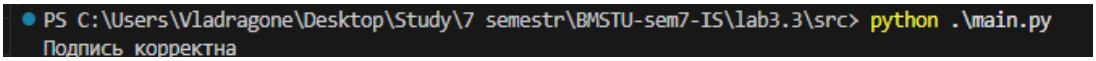

Рисунок 4.5 – Совпадение подписи

А если попробуем сравнить подпись с другим архивом:

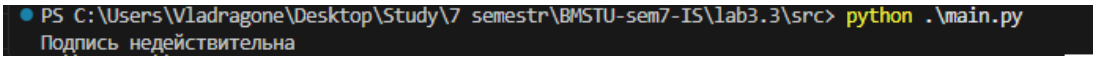

Рисунок 4.6 – Не совпадение подписи