НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра Інформаційної Безпеки

Лабораторна робота №4 дисципліни

”КРИПТОГРАФІЯ”

Підготував:

студент групи ФБ-03

Заболотний Максим

Київ 2023

Тема роботи: Вивчення криптосистеми RSA та алгоритму електронного підпису; ознайомлення з методами генерації параметрів для асиметричних криптосистем.

Мета роботи: Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.

Порядок виконання роботи

1. Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч, використання готових реалізацій тестів не дозволяється.
2. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p1, q1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq p1q1 ; p і q – прості числа для побудови ключів абонента A, 1 p і q1 – абонента B.
3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p,q) та відкритий ключ (n,e) . За допомогою цієї функції побудувати схеми RSA для абонентів A і B – тобто, створити та зберегти для подальшого використання відкриті ключі (e,n) , (,) 1 n1 e та секретні d і d1 .
4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів A і B. Кожна з операцій (шифрування, розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті ключові дані, які необхідні для її виконання. За допомогою датчика випадкових чисел вибрати відкрите повідомлення M і знайти криптограму для абонентів A и B, перевірити правильність розшифрування. Скласти для A і B повідомлення з цифровим підписом і перевірити його.
5. За допомогою раніше написаних на попередніх етапах програм організувати роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA. Протоколи роботи кожного учасника (відправника та приймаючого) повинні бути реалізовані у вигляді окремих процедур, на вхід до яких повинні подаватись лише ті ключові дані, які необхідні для виконання. Перевірити роботу програм для випадково обраного ключа 0 k n.

Хід роботи

1. Пошук випадкового простого числа та його перевірка на простоту тестом Міллера-Рабіна:

```python
def random_number(start, finish):
    def miller_rabin(number):
        if number % 2 == 0 or number % 3 == 0 or number % 5 == 0 or number % 7 == 0 or number % 11 == 0 or number % 13 == 0:
            return False
        d, s = number - 1, 0
        while True:
            if d % 2 == 0:
                break
            d = d // 2
            s += 1
        rand = random.randint(2, number - 2)
        if gcd(rand, number) > 1:
            return False
        elif pow(rand, d, number) == 1 or pow(rand, d, number) == -1:
            return True
        else:
            for i in range(1, s - 1):
                rand = pow(rand, 2, number)
                if rand == - 1:
                    return True
                if rand == 1:
                    return False
            return False

    while True:
        number = random.randint(start, finish)
        if miller_rabin(number) is True:
            return number
```

2. Генеруємо дві пари простих чисел p, q та p1,q1:

```python
while True:
    sender = (random_number(2 ** 256, 2 ** 260), random_number(2 ** 256, 2 ** 260))
    recipient = (random_number(2 ** 256, 2 ** 260), random_number(2 ** 256, 2 ** 260))
    if sender[0] * sender[1] > recipient[0] * recipient[1]:
        break


def gen_keys(p, q):
    n = p * q
    f = (p - 1) * (q - 1)
    e = 2 ** 16 + 1
    d = pow(e, -1, f)
    open = (e, n)
    close = (p, q, d)
    return open, close


sender_open, sender_close = gen_keys(sender[0], sender[1])
recipient_open, recipient_close = gen_keys(recipient[0], recipient[1])
message = random.randint(2 ** 256, 2 ** 260)
```

3. Програма шифрування:

```
sender_open, sender_close = gen_keys(sender[0], sender[1])
recipient_open, recipient_close = gen_keys(recipient[0], recipient[1])
message = random.randint(2 ** 256, 2 ** 260)


def encryption(message, sender_ok):
    encrypted_message = pow(message, sender_ok[0], sender_ok[1])
    return encrypted_message


encrypted_message = encryption(message, recipient_open)
```

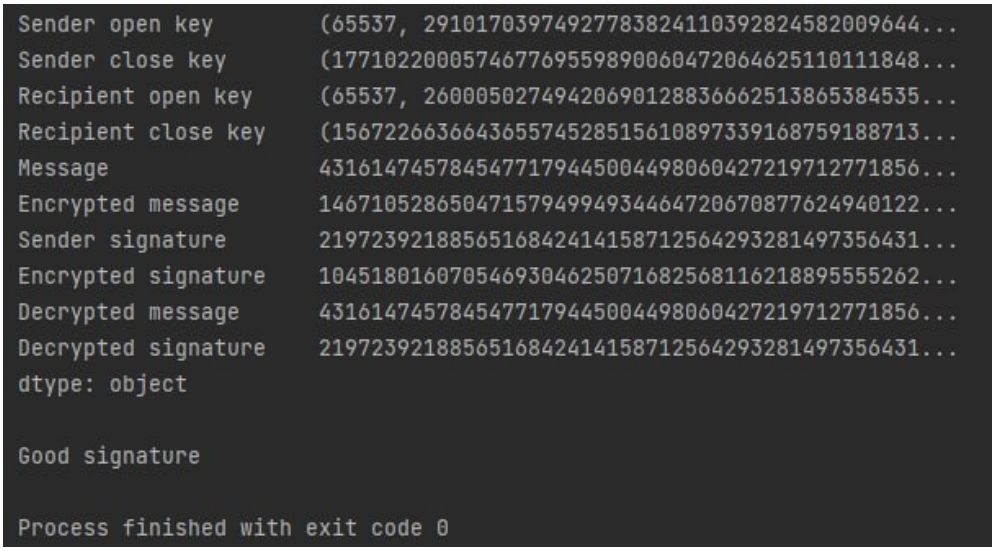
Програма розшифрування:

```
def decryption(message, recipient_ok):
    decrypted_message = pow(message, recipient_ok[2], recipient_ok[0] * recipient_ok[1])
    return decrypted_message


decrypted_message = decryption(encrypted_message, recipient_close)
decrypted_signature = decryption(encrypted_signature, recipient_close)
```

Створення цифрового підпису:

```
def signature_creation(message, sender_ck):
    signature = pow(message, sender_ck[2], sender_ck[0] * sender_ck[1])
    return signature


signature = signature_creation(message, sender_close)
encrypted_signature = encryption(signature, recipient_open)
```

5. Результат виконання:



Висновок

Виконуючи даний лабораторний практикум, я ознайомився з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практично ознайомився з системою захисту інформації на основі криптосхеми RSA, організував з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.