КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №5

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

Мета та основні завдання 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.

Спочатку написала тест Міллера-Рабіна.

```
def MillerRabin(n, k):
    if n == 2:
        return True

    if n % 2 == 0:
        return False

    r, s = 0, n - 1
    while s % 2 == 0:
        r += 1
        s //= 2
    for i in range(k):
        a = random.randrange(2, n - 1)
        x = pow(a, s, n)
        if x == 1 or x == n - 1:
            continue
        for j in range(r - 1):
            x = pow(x, 2, n)
            if x == n - 1:
                break
        else:
            return False
    return True
```

Також в майбутньому нам знадобляться функції з 3 лаби.

Зробила генератор рандомних чисел за інсрукцією у лабі, для більшої безпеки генеруються числа виду p=2ip+1

```
def GenerateQP(n):
    p = random.randint(n, 2*n-2)
    while(MillerRabin(p, 20)!=True):
        p = random.randint(n, 2*n-2)

    q = random.randint(n, 2*n-2)
    while (MillerRabin(q, 20) != True):
        q = random.randint(n, 2*n - 2)

    i = 1
    p = 2*i*p+1
    while (MillerRabin(p, 20)!=True):
        i = i+1
        p = 2 * i * p + 1

    j = 1
    q = 2 * j * p + 1
    while (MillerRabin(q, 20) != True):
        j = j + 1
        q = 2 * j * q + 1

    return (p, q)
```

При побудові генератора ключів, використала попередню функцію. Функція виводить public(n, e) та privat(p, q, d)

```
def GenerateKayPair(x):
    s = GenerateQP(x)
    p = s[0]
    q = s[1]
    n = p*q
    o = (p-1)*(q-1)
    e = 65537
    d = rivnyanya(e, 1, o)
    public = (n, e)
    privat = (p, q, d)
    return public, privat
```

Написала короткі функції для шифрування, розшифрування і створення повідомлення з цифровим підписом.

Поділила код на А и В. А відправляє повідомлення В.

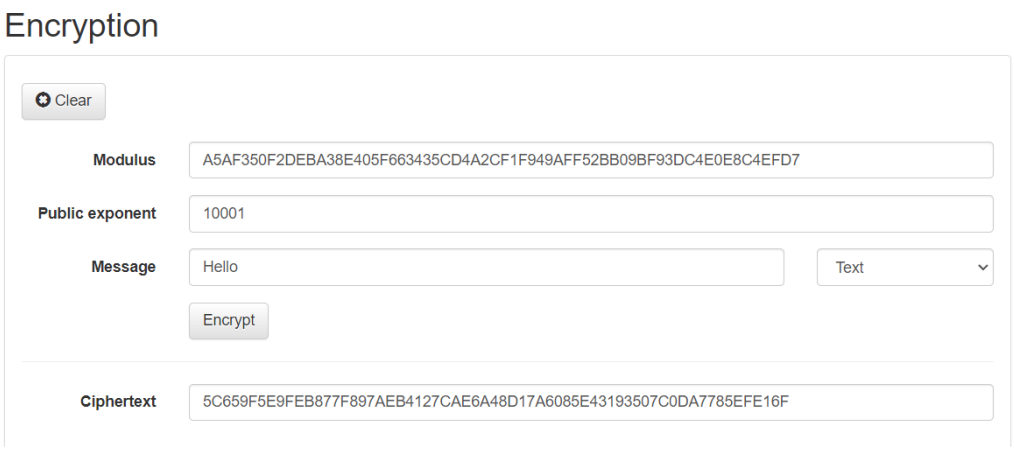
```python
#A abonent
x_a = GenerateKayPair(128)
public_a = x_a[0]
privat_a = x_a[1]
n = public_a[0]
e = public_a[1]
d = privat_a[2][0]
print(n, e, d)

#B abonent
x_b = GenerateKayPair(129)
while (x_b[0][0]<n):
    x_b = GenerateKayPair(129)
public_b = x_b[0]
privat_b = x_b[1]
n1 = public_b[0]
e1 = public_b[1]
d1 = privat_b[2][0]
print(n1, e1, d1)

#A abonent
k = int(input('A number'))
message_a = SendKey(k, n, n1, e1, d)

#B abonent
encrypt_a = ReceiveKey(message_a[0], message_a[1], d1, n, e, n1)
print(encrypt_a)
```

Вивід для програми:

```
266264093 65537 237640373
1245175594283 65537 3770020769
A number23
23
```

Намагалась перевірити у програмі по посиланню.

Перевірила функцію Encrypt()



Далі почались проблеми, бо моя програма не підтримує текст, лише числа.

Висновок: ознайомилась з тим, як працює перевіркана прості числа та криптосистема RSA.