Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Виконали:

студенти III курсу ФТІ групи ФБ-96
Шидлюх Максим та Шафрай Ілля

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.

Завдання

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 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб 1 1 pq p q ; p і q – прості числа для побудови ключів абонента A, 1 p і 1 q – 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 1 e n та секретні d і 1 d .
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.
Кожна з наведених операцій повинна бути реалізована у вигляді окремої процедури, інтерфейс якої повинен приймати лише ті дані, які необхідні для її роботи; наприклад, функція Encrypt(), яка шифрує повідомлення для абонента, повинна приймати на вхід повідомлення та відкритий ключ адресата (і тільки його), повертаючи в якості результату шифротекст. Відповідно, програмний код повинен містити сім високорівневих процедур: GenerateKeyPair(), Encrypt(), Decrypt(), Sign(), Verify(), SendKey(), ReceiveKey().

Тест Міллера-Рабіна

```
def miller_rabin(n, k=4):
    if n < 3: return False
    for p in [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]:
        if n % p == 0:
            if n == p: return n
            else: return False
    s, d = 0, n - 1
    while d % 2 == 0:
        s, d = s + 1, d // 2
    for i in range(k):
        x = pow(random.randint(2, n - 1), d, n)
        if x == 1 or x == n - 1: continue
        for r in range(1, s):
            x = (x * x) % n
            if x == 1:return False
            if x == n - 1: break
        else: return False
    return n
```

Генерація випадкових простих чисел

```
def generate_key(a):
    while True:
        n = random.randint(a,a*2)
        p = miller_rabin(n)
        if p is not False:break
    return p

def generate_pq():
    p, q = generate_key(a), generate_key(a)
    p1, q1 = generate_key(a), generate_key(a)
    while p * q > p1 * q1:
        p, q = generate_key(a), generate_key(a)
        p1, q1 = generate_key(a), generate_key(a)
    return p, q, p1, q1
```

скріншоти виконання роботи:

```
p*q 46329348190303446071429438206409352901635920163839391962197739202608487553804924320396894816994911593904235235973806929852549166486691140218283004410521867
p1*q1 50441695109160108446916831697597822295648855270848560377847406467244490467973694633967982042082759400596565887479700332872249094991838847304655992993973487
m= 11036817130343381555791025324766286427209952653082275949168507275331977725146 7
Massage sended [10170168324988378997131054359635641927304994794904721163053874118025739114662471504207675025270543497483115856672319435473592687512841936004524015226768627,
 17396580930562477527946389007179499744948337009269859796339381640883924743850688482292329574473196070407106608299668613110241865916415108024935400606477784]
Massage received [11036817130343381555791025324766286427209952653082275949168507275331977725146 7, True]
```

Скріншоти роботи з сайту

```python
#Server tests

open_test=["10001","9D5E56535B05820AD9D17C1AD1CBEEAEFEE2AD5A757C6B155DCE6C4B5EB7524B"]
print("Open key",open_test)
open_test[0],open_test[1]=int(open_test[0], 16),int(open_test[1], 16)
M_test="text to penis"
print("Message '"+ M_test+"'")
M_test = int(M_test.encode().hex(), 16)
C_test=encrypt(M_test,open_test)
print("Ciphertext",hex(C_test)[2:])
sign="77B3F041D5663A222A723D3BE708E7CA64C99CF3AFAAA0161727C4F5FED536DD"
sign=int(sign, 16)
print("sign is", M_test==encrypt(sign, open_test))
```

Get server key

Clear

Key size 256

Get key

Modulus 9D5E56535B05820AD9D17C1AD1CBEEAEFEE2AD5A757C6B155DCE6C4B5EB7524B

Public exponent 10001

Encryption

Clear

Modulus 9D5E56535B05820AD9D17C1AD1CBEEAEFEE2AD5A757C6B155DCE6C4B5EB7524B

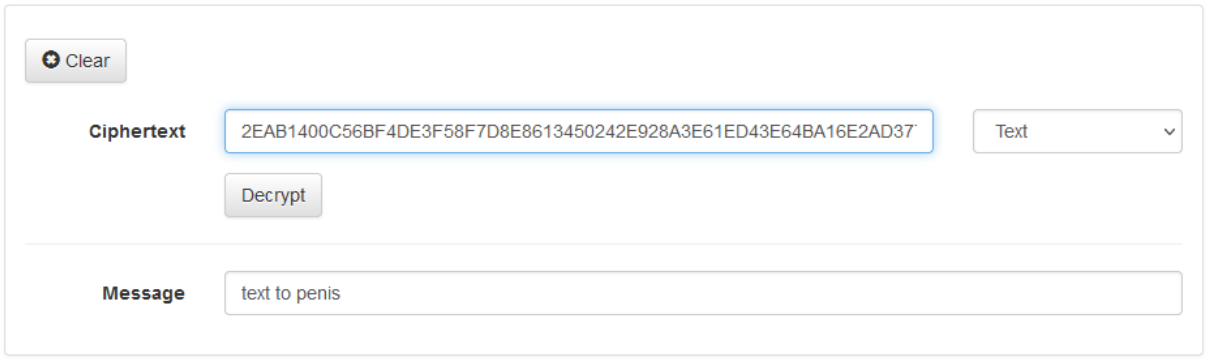
Public exponent 10001

Message text to penis Text

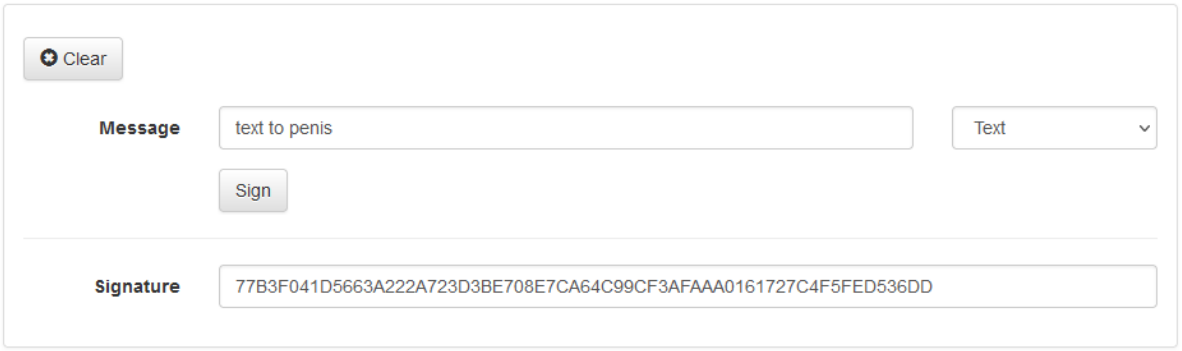
Encrypt

Ciphertext 2EAB1400C56BF4DE3F58F7D8E8613450242E928A3E61ED43E64BA16E2AD37738

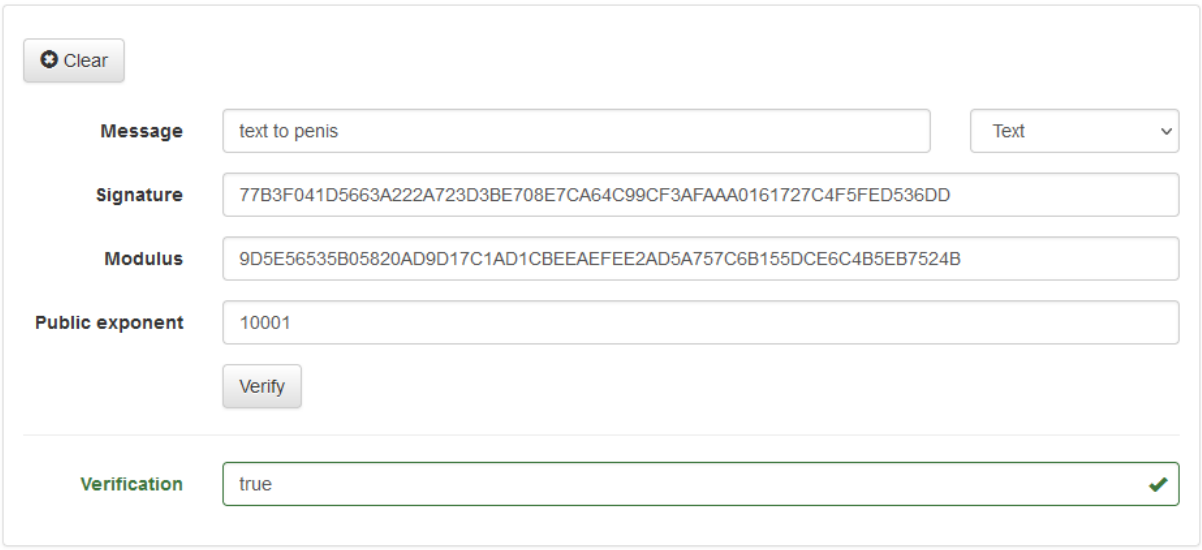
Decryption



Sign



Verify




```
Open key ['10001', '9D5E56535B05820AD9D17C1AD1CBEEAEFEE2AD5A757C6B155DCE6C4B5EB7524B']
Message 'text to penis'
Ciphertext 2eab1400c56bf4de3f58f7d8e8613450242e928a3e61ed43e64ba16e2ad37738
sign is True
```

Висновок:

Ознайомились 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 засекречений зв'язок й електронний підпис, вивчив протокол розсилання ключів.