НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

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

Виконали:
студенти групи ФБ-22
Шафранський Данііл
Перевузник Ілля

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.

Порядок і рекомендації щодо 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.

```python
import random

def zgheneruvaty_proste_chyslo(dovzhyna_bita):
    while True:
        chyslo_kandydat = random.getrandbits(dovzhyna_bita) | (1 << dovzhyna_bita - 1) | 1
        if prostye_chyslo(chyslo_kandydat):
            return chyslo_kandydat
        else:
            print(f"Kandydat {chyslo_kandydat} ne proste chyslo.")

def vyznachennya_nsd(a, b):
    while b:
        a, b = b, a % b
    return a

def prostye_chyslo(chyslo, iteratsii=5):
    if chyslo <= 1 or chyslo % 2 == 0:
        return False
    if chyslo == 2:
        return True

    eksponenta, zalysok = 0, chyslo - 1
    while zalysok % 2 == 0:
        eksponenta += 1
        zalysok //= 2

    for _ in range(iteratsii):
        baza = random.randint(2, chyslo - 2)
        kandydat = pow(baza, zalysok, chyslo)
        if kandydat == 1 or kandydat == chyslo - 1:
            continue

        for _ in range(eksponenta - 1):
            kandydat = pow(kandydat, 2, chyslo)
            if kandydat == chyslo - 1:
                break
        else:
            return False
    return True
```

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 pq p1q1 ; p і q – прості числа для побудови ключів абонента А, 1 p і q1 – абонента В.

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 А і В – тобто, створити та зберегти для подальшого використання відкриті ключі (e,n) , (,) 1 n1 e та секретні d і d1 .

```python
def zgheneruvaty_kluchovi_pary(dovzhyna_bita):
    chyslo_p = zgheneruvaty_proste_chyslo(dovzhyna_bita)
    chyslo_q = zgheneruvaty_proste_chyslo(dovzhyna_bita)

    while chyslo_p == chyslo_q:
        print(f"Chysla p i q odnakovi. Perhe nerelytuvaty q.")
        chyslo_q = zgheneruvaty_proste_chyslo(dovzhyna_bita)

    modul_n = chyslo_p * chyslo_q
    totyent_phi = (chyslo_p - 1) * (chyslo_q - 1)

    while True:
        publichnyy_eksponent_e = random.randint(3, totyent_phi - 1)
        if vyznachennya_nsd(publichnyy_eksponent_e, totyent_phi) == 1 and prostye_chyslo(publichnyy_eksponent_e):
            break

    private_eksponent_d = modularna_oborotna(publichnyy_eksponent_e, totyent_phi)

    return {'publichnyy_kluch': (publichnyy_eksponent_e, modul_n), 'private_kluch': (private_eksponent_d, chyslo_p, chyslo_q)}
```

4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і В. 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 М і знайти криптограму для абонентів А и В, перевірити правильність розшифрування. Скласти для А і В повідомлення з цифровим підписом і перевірити його.

```python
def shifruvaty(plaintext, publichnyy_kluch):
    publichnyy_eksponent_e, modul_n = publichnyy_kluch
    return pow(plaintext, publichnyy_eksponent_e, modul_n)

def rozshifruvaty(ciphertext, private_kluch):
    private_eksponent_d, chyslo_p, chyslo_q = private_kluch
    modul_n = chyslo_p * chyslo_q
    return pow(ciphertext, private_eksponent_d, modul_n)

def pidpysaty(plaintext, private_kluch):
    private_eksponent_d, chyslo_p, chyslo_q = private_kluch
    modul_n = chyslo_p * chyslo_q
    return pow(plaintext, private_eksponent_d, modul_n)

def pereviryty(pidpys, publichnyy_kluch):
    publichnyy_eksponent_e, modul_n = publichnyy_kluch
    return pow(pidpys, publichnyy_eksponent_e, modul_n)
```

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.

```
def nadyslaty_kluch(spilnyy_kluch, publichnyy_kluch_otrymuvacha):
    return shifruvaty(spilnyy_kluch, publichnyy_kluch_otrymuvacha)

def otrymaty_kluch(enkryptyvanyy_spilnyy_kluch, private_kluch_otrymuvacha):
    return rozshifruvaty(enkryptyvanyy_spilnyy_kluch, private_kluch_otrymuvacha)
```

– чисельні значення прикладів ВТ, ШТ, цифрового підпису;

```
Randomne povidomlennya:
708516558473563841863384312744532030489563695621461371888141708522982119371939
304581336290684485423411977638781088237612225799193631445323444502090081338 9
Enkryptyvanyy dlya Korysuvacha A:
628108628607553186766435494836604384190572761465564360230421714756302922590530
359827301629437281664597713449758356136864652866132894121078137237125675719 6
Rozshifruvanyy dlya Korysuvacha A:
708516558473563841863384312744532030489563695621461371888141708522982119371939
304581336290684485423411977638781088237612225799193631445323444502090081338 9
Enkryptyvanyy dlya Korysuvacha B:
296130882080583533352065449835662679387504606036709504749355957066761932895100
309634006684390171442930801627615872943683262066931441747739973723993331927 6
Rozshifruvanyy dlya Korysuvacha B:
201306563649082089299486322641654243545342148837594098486408163662020281858710
259456159384047355971100839108356472910274799582706270060575321893321664632 6
```

– опис кроків протоколу конфіденційного розсилання ключів з підтвердженням справжності, чисельні значення характеристик на кожному кроці:

1. Для абонентів A та B генеруються прості числа p, q для A і p1, q1 для B.
   p та q -
   ```
   112257917030563949810756103070873017160681537692602498766415888519618069412423,
   683722401834739456355001517878381131972654264733629612336074257362457238150 51
   ```
   p1 та q1 -
   ```
   598508234542909727713096468596383590307230175506983294317829361654156716327 37,
   847457003180325672376147545677521334418055844368235509838976622945680404653 99
   ```

2. На основі цих чисел для абонентів A та B обчислюються відкриті ключі (n, e) для A і (n1, e1) для B, а також секретні ключі d і d1 відповідно.
   (n, e) -
   591938017677368900044288293511878575936984643703045435119046725450129222229247427441651415007055383365368872845277963515294239093619072515790870564756041 7,
   767532526571020867822914761949781986678052005970035894057966425693124243810458363884102138536911500660461593129112734330095382729004159299178390949377857 3
   (n1, e1) -
   456505476990759359599238142084103013716805827398923011829840678587666915314011008919156357022759102299896838139097653782679900099544834536488979140238713 3,
   507209994824481752563897990102877786944221546783867273401733544860961837513229045125176906637129452311138530424615327337426216487361384748122608768416706 3
   d -
   861851364486976875666759173086473977982238229042393561158592876042014340238265754023954155849883476076032943052779612754873417354209330480253578645001053
   d1 -
   397754433205138752740370514887314043882387361477434905790341239856373838951429781900721724094826354265736832841758956239452951583477741181852277772574952 5
3. Абонент A шифрує своє повідомлення, використовуючи відкритий ключ абонента B (n1, e1).
   Randomne povidomlennya:
   708516558473563841863384312744532030489563695621461371888141708522982119371939304581336290684485423411977638781088237612225799193631445323444502090081338 9 Enkryptyvanyy dlya Korysuvacha A:
   628108628607553186766435494836604384190572761465564360230421714756302922590530359827301629437281664597713449758356136864652866132894121078137237125675719 6
4. Абонент A створює цифровий підпис повідомлення за допомогою свого секретного ключа (d), після чого шифрує підпис, використовуючи відкритий ключ абонента B (n1, e1).
   Tsifrovyy pidpys Korysuvacha A:
   266356049308274701826509377690695423986050135829253539967358695222425209762181602731124105448033911944030613027962634876780825910641720221139774946755613 1
5. Абонент A передає зашифроване повідомлення (m) та зашифрований цифровий підпис (s) абоненту B.

6. Абонент B розшифровує отримане повідомлення, використовуючи свій секретний ключ (d1).

7. Абонент B розшифровує цифровий підпис: спершу своїм секретним ключем (d1), а потім відкритим ключем абонента A (n, e).

8. Абонент B перевіряє достовірність підпису, порівнюючи його з розшифрованим повідомленням.
   ```
   Perevirka kluchiv vid Korysuvacha A do Korysuvacha B... Peredavanyy kluch: 626 Otrymanyy kluch: 626
   ```

Перевірка на сайті:
Encryption:

Encryption

Clear

Modulus 5F6F4942B76B5DE507EE8B80953D35402268864EE17A1B43D4'

Public exponent 10001

Message 202B5576E92ECDCDC4109744807C6732FF3C Bytes

Encrypt

Ciphertext 12D753EB317F0A4A360C2FE83889333672876A34B24A445574A

From Hexadecimal

To Decimal

Enter hex numbers

12D753EB317F0A4A360C2FE838893336728 16

= Convert × Reset ⇅ Swap

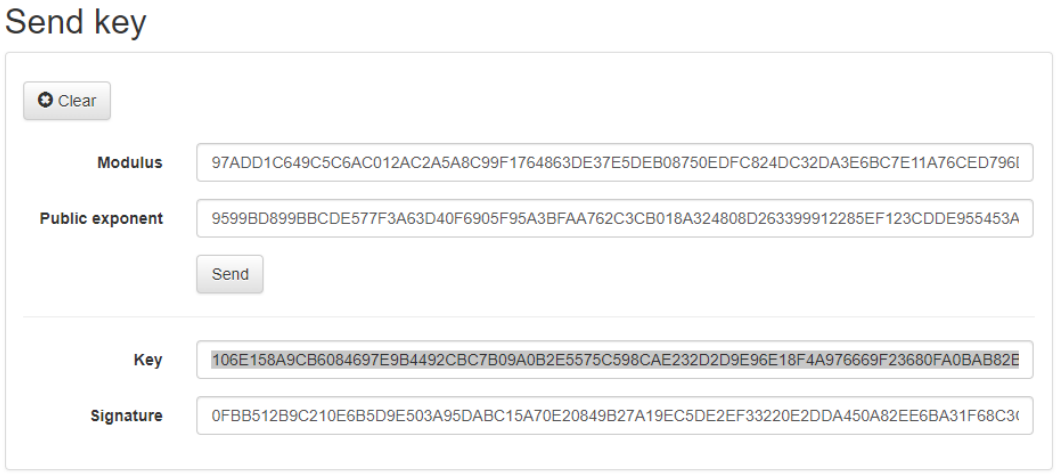
Decimal number (153 digits)

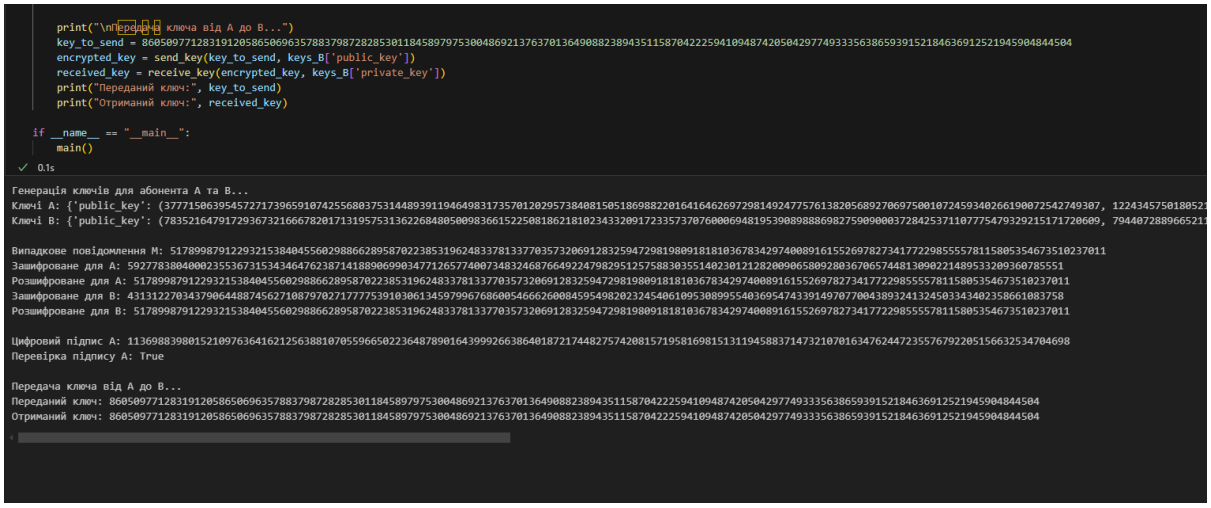
```
986789746495373332543481015220467274342
270408929334271258907680444713035588377
586800741457355808641021448051700750276
691050389442963018293645001943747431
```

10

```
Випадкове повідомлення M: 16848415289598377487750810566582256933436525316456647822452769581811494820481624888239285366968640647624465365782508117975202410405293943856723078527580360
Зашифроване для A: 986789746495373332543481015220467274342704089293342712589076804447130355883775868007414573558086410214480517007502766910503894429630182936450019437474310
```

Key send:





Висновки

Ознайомилися з методами перевірки чисел на простоту та алгоритмами генерації ключів для асиметричних криптосистем типу RSA. Практично дослідили застосування криптосистеми RSA для захисту інформації, включаючи організацію засекреченого зв'язку та створення електронного підпису. Вивчили протокол розповсюдження ключів у межах цієї системи.