Протокол лабораторної роботи №4

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

Варіант №7

Виконав

Студент 3 курсу

Групи ФБ-13

Короткевич Іван

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.

Код програми:

Створення 256 битного числа та перевірка числа на простосту:

```
#Generate random 526 bit number
def number_256bit():
    random_number = secrets.randbits(256)
    return random_number


#Function to check if number is prime
def check_prime(p, k=15):

    if p % 2 == 0 or p % 3 == 0 or p % 5 == 0 or p % 7 == 0:
        return False

    s = 0
    d = (p - 1)
    while d % 2 == 0:
        d //= 2
        s += 1

    for i in range(k):
        x = randrange(2, p - 1)

        if gcd(x, p) > 1:
            return False

        a = pow(x, d, p)
        if a == 1 or a == -1:
            continue

        for _ in range(1, s):
            a = pow(a, 2, p)

            if a == p - 1:
                break

        else:
            return False

    return True
```

Знаходження простого чилса:

```python
def select_prime(number):

    if number % 2 == 0:
        x = number + 1
    else:
        x = number

    i = 1
    while i < number / 2:
        if check_prime(x) == True:
            return x

        x += 2*i
        i += 1

    return x
```

Шифрування, дешифрування, створення підпису та перевірка повідомлення:

```python
def GenerateKeyPair(p, q):
    n = p*q
    phi_n = (p-1)*(q-1)
    exp = secrets.randbits(16)
    e = select_prime(exp)
    d = inverse_elem(e, phi_n)
    return n, e, d


#Encrypt message
def Encrypt(M, e, n):
    C = pow(M, e, n)
    return C


#Sign the message
def Sign(M, d, n):
    S = pow(M, d, n)
    return S


#Descrypt the message
def Decrypt(C, d, n):
    M = pow(C, d, n)
    return M


#Verify integrigty of the message
def Verify(M, S, e, n):
    if M == pow(S, e, n):
        return True
    return False
```

Відправлення та отримання ключа:

```python
#Function that sends common secret
def SendKey(k, d, e1, n, n1):
    k1 = pow(k, e1, n1)
    S = pow(k, d, n)
    S1 = pow(S, e1, n1)
    return k1, S1


#Function that recieves common secret
def RecieveKey(k1, S1, d1, n1):
    k = pow(k1, d1, n1)
    S = pow(S1, d1, n1)
    return k, S
```

Приклад роботи програми:

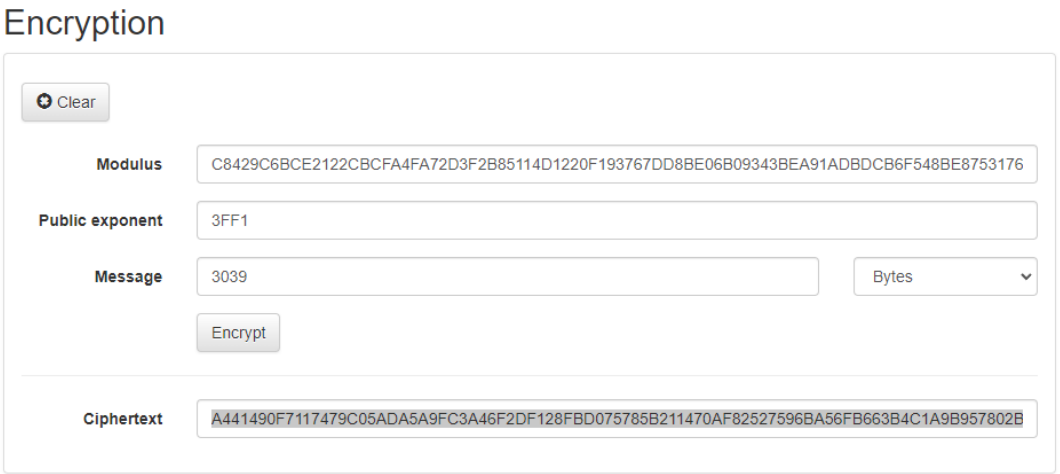
```
Ciphertext for B:  57063827247076009810695030921508137434513230017667915687868718798664433372017232159374579513387360169248264937547757975043685503015573515914898079635862 5
e1:  39937
Modulus1:  769625199170794701664345553169374517444442819983028145370202946236470436559336066352585573743331372057053679329514732209354947197599957741757940472843932 1
Signature for b:  42902871052326726444313267348625031507551592211868510303600892215484102000914651298743880928622353017753222479498235948477260422493757545562462013493819
Ciphertext for A:  30142657851040080403183369005828688061252384038639657754592497139746706620657475126388072380955103812184749150918832095891417222812875033139304196271103
e:  42689
Modulus:  655640962739028238840599562027339325796221088576861325835032886034528101655913838256530806762155613106927771147493244796669436642974666839698766886434 7
Signature for b:  31061669925197973634780208231901443000835731654661860987363635069664105612043105327969358227905195896432854758928616698661669376425961765101494136679667 6
Verification result for B:  True
e:  42689
Modulus:  655640962739028238840599562027339325796221088576861325835032886034528101655913838256530806762155613106927771147493244796669436642974666839698766886434 7
Plaintext:  67890
Verification result:  True
e1:  39937
Modulus1:  769625199170794701664345553169374517444442819983028145370202946236470436559336066352585573743331372057053679329514732209354947197599957741757940472843932 1
Plaintext:  12345
k1 3562664273455145991909602098377691544669562679311109864406255789411558083099910800066182922395502078220384776350278161902876403695027445835412972167348743
S1:  64465228681205818257121649577235164601800145574537934370570361570455818267519375737573436693127742687034905515242991615339201955653002394894118790391362 0
k:  12345
Verification result:  True
```

Програма може зберігати ключі у спеціальному файлі keys.json

Перевірка роботи:

Повідомлення: 12345

Для перевірки роботи значення треба було перевести в 16-ти систему счислення:



```
[Running] python -u "c:\Users\Vano\OneDrive\Рабочий стол\Uni\Crypto\lab4\lab4.py"
Ciphertext:  860273349190625913308639109244183710734853468724352380278201252908374304542781132148739369053972451200875721269593425040680426527597750556659850771807728 0
e1:  16369
Modulus:  104884776882160363090443663389970689854210753085158145824545147549252916440731096417404247155594605046256776212097111791327077228152484262651696406892371 21
Signature:  155614374070000789162628170233881754592534813613242799587263231918190272436308955486813768542948631786489078893352237851349418049258775479296581797333228

[Done] exited with code=0 in 0.068 seconds
```

Hexadecimal to Decimal converter

From

Hexadecimal

To

Decimal

Enter hex number

A441490F7117479C05ADA5A9FC3A46F2DF

16

= Convert × Reset ⇅ Swap

Decimal number (154 digits)

860273349190625913308639109244183
710734853468724352380278201252908

10


```
{
  "p": 165639576754751596407827317983292449101826783239770375966606646299543800140 81,
  "q": 301013613263403295622475005569349771715832953029614552284218040288936075788 9,
  "p1": 940109068306712078028594804134082410856555126308739677747792114143068157440 49,
  "q1": 111566604788819607361844769735839102883821668689541296669805543799233150805 329,
  "e": 50833,
  "e1": 16369,
  "d": 194993838228624679774247783729169413315879346335456240118611788497992925491058987981324972168692626902072979665906039311883447455093878287185387452086897,
  "d1": 100335434125343583947294112347947402555262153006322402435491017500992816821148270964480946309652958603506872018060754664827890432487040964142221626987945 13,
  "n": 498597674983686033549513963295013520477167747096088886013550957983826679149292215422356559465619142069029095980482253717136560448800289459227002651835009,
  "n1": 104884776882160363090443663389970689854210753085158145824545147549252916440731096417404247155594605046256776212097111791327077228152484262651696406892371 21
}
```

Verify

Clear

Message 3039 Bytes

Signature 2F8A0913B1B74A581C015076190C3E16759C339DAF3D0DB8CABD8DCF88F712510CACCFDB34A6BAF4BCB

Modulus 985183C389FE41F472644F36BF5AB76A2771194CB1DE00A3ABA980A0CC87FA32A78F185611605CFD9CB0A

Public exponent C691

Verify

Verification true

```
[Running] python -u "c:\Users\Vano\OneDrive\Рабочий стол\Uni\Crypto\lab4\lab4.py"
Ciphertext:  860273349190625913308639109244183710734853468724352380278201252908374304542781132148739369053972451200875721269593425040680426527597750556659850771807728 0
Signature:  155614374070000789162628170233881754592534813613242799587263231918190272436308955486813768542948631786489078893352237851349418049258775479296581797333228
Verification result:  True
e:  50833
Modulus:  498597674983686033549513963295013520477167747096088886013550957983826679149292215422356559465619142069029095980482253717136560448800289459227002651835009
Plaintext:  12345

[Done] exited with code=0 in 0.069 seconds
```