Міністерство освіти і науки України Національний технічний університет України "Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

КРИПТОГРАФІЯ КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

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

Виконав:
Дворніков Дмитро
Варіант 8
Група:
ФБ-03

Київ - 2022

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.
Хід 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.

У якості тесту на простоту використовував рекомендований тест Міллера-Рабіна, також написав свю функцію для піднесення до степеня. Алгоритм степенної функціїї створений на основі схеми Горнера, яка розглядалась у методичці. Також алгоритм знаходження випадкового числа робилося завдякі інтервалу від 2**255 до 2*256

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 A, 1 p і q1 – абонента B.

Ось такі ключі для абонентів A та B в мене вийшли:

```
59051682366422182621922571724099874503066455249210652413330759674974193547107
83110053063839336363642961537879308914517500279516609283409194923737635655899
89983359773739261877837346127661223468012102002752881713618655567656480931567
85172386467276780140160506814992678941396763599639592196904568023806745057507
```

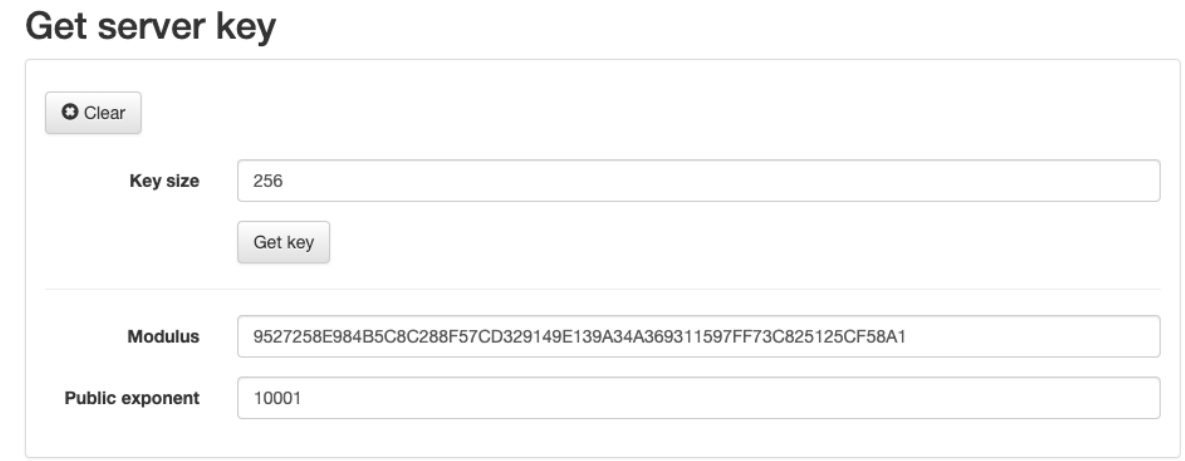
3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ (d, p,q) та відкритий ключ (n,e) . За допомогою цієї функції побудувати схеми RSA для абонентів A і B – тобто, створити та зберегти для подальшого використання відкриті ключі (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.

Ось приклад працездатонсті мого коду, спочатку перевірка шифрування/дешифрування, потім перевірка підписки, потім первірка отримання ключа.

```
Message - 976512771455649133483863004489539459818013249624573599186063639220655429528 60
Encrypt message - 766796272460558621491246738569950206862992316713573127237405341443477576196781116560
Decrypt message - 976512771455649133483863004489539459818013249624573599186063639220655429528 60
True signature
Key k - 106479588209892663361778936045052048345653246684287109021813915057035385402801380053662290996 3
Key k1,s1 - 4700509147158771230959317752994066567834107354718544468512245463314626291730431714521071 20
User auth 106479588209892663361778936045052048345653246684287109021813915057035385402801380053662290 99
```

Перевірка завдяки сайту:



Encryption

Clear

Modulus 9527258E984B5C8C288F57CD329149E139A34A369311597FF73C825125CF58A1

Public exponent 10001

Message 03 Bytes

Encrypt

Ciphertext 384C6326160C3741138C719E936B54D6850102BB7D6D393822277741DFA78903

```
# mod 9527258E984B5C8C288F57CD329149E139A34A369311597FF73C825125CF58A1
n = 67463780683398777104889340847676156594341795278182954262458502390843331008673
# public exp 10001
exp = 65537

web = CryptoSystem("web",exp,n,d=None)

#  message 03
message = 3
#
# encrypt 384C6326160C3741138C719E936B54D6850102BB7D6D393822277741DFA78903
encrypt = web.encrypt(message)
print(encrypt)
```

Encpypt message :

$$2546448419727427629595735107944799171677170546669707458867889940086591111398 7_{10}$$

Исходное основание
10
Основание системы счисления исходного числа

Исходное число
2546448419727427629595735107944799171677170546
Число которое необходимо преобразовать

Основание результата
16
Основание системы счисления переведенного числа

Переведенное число
384C6326160C3741138C719E936B54D6850102BB7D6D393822277741DFA78903

Decryption

Clear

Ciphertext 384C6326160C3741138C719E936B54D6850102BB7D6D393822277741DFA78903 Bytes

Decrypt

Message 03

Sign

Clear

Message 03 Bytes

Sign

Signature 6C460F54C0E3CF026B3523076285E1D029F566040187245AF42F83E5D9A782DA

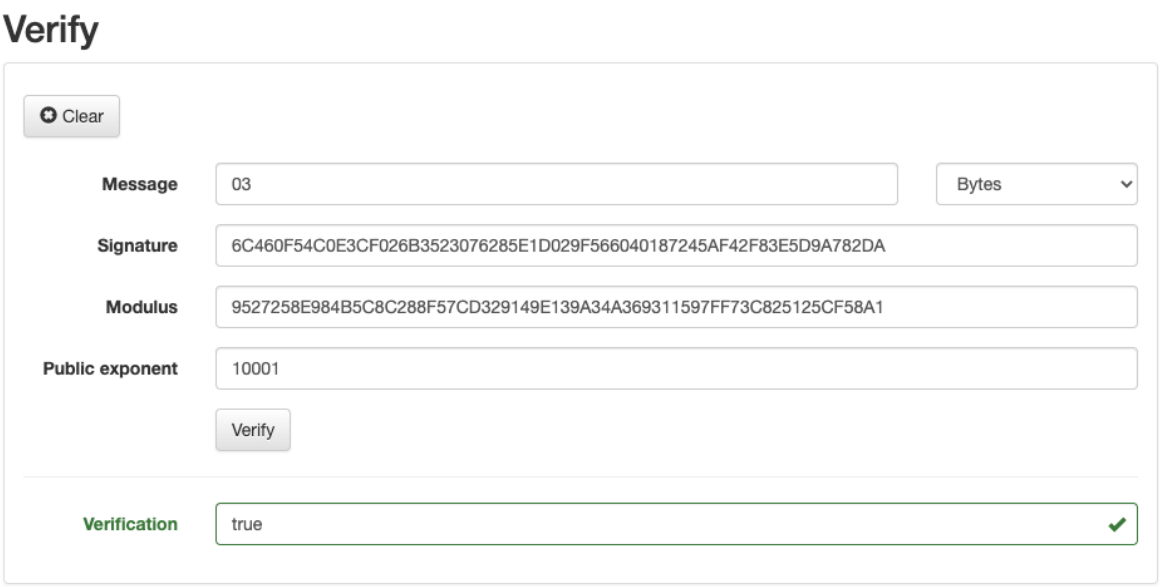
```
# signature 6C460F54C0E3CF026B3523076285E1D029F566040187245AF42F83E5D9A782DA
sign = 489735727526716537818494396329015401112467581282075194334392336808866831080 58
#
verify = web.check_sign(message,sign)
print(str(verify),"verification status")
```

Исходное основание
16
Основание системы счисления исходного числа

Исходное число
6C460F54C0E3CF026B3523076285E1D029F566040187
Число которое необходимо преобразовать

Основание результата
10
Основание системы счисления переведенного числа

Переведенное число
489735727526716537818494396329015401112467581282075194334392336808866831080 58




```
True verification status
```

Висновки

Завдяки цієї роботі я зрозумів як працює один з найбільш важливих кристосистем а васме криптосистема RSA. Отримав практичні навички у перевірці великих чисел на простоту також, поліпшив розуміння роботи криптосистеми. Завдяки онлайн перевірці переконався у правильності мною на