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

Виконали:

Акент'єв Влад, Шапоренко Микита

Група: ФБ-06

Київ – 2022

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

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.

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 $\leq$ p1q1 ; p і q – прості числа для побудови ключів абонента A, 1 p і q1 – 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 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 $0 < k < n$.

Результати

Message : 550

-------A-------

Public exponent: 100001

Public n:
193163445556798214071035565830998378119882870795349654125894782207649702712492973646287478800977438515724102992660416055129078751119942090221775328312118 01

Encrypted A:
817104273926012532834567864723316705779394294761203712004699049242627799352230204291001594342137784750492316509518667947784972445960113033754970398443222 8

Decrypted A: 550

Sign A:
191477453448464079136288669792846506074603234839691323277164544477721995206813953055721883698958108176612167761552631951270912115563313932539969186907012 26

Verify A: True

-------B-------

Public exponent1: 100001

Public n1:
472785812706825727016533937894309437546514708893101940998385640590145305930194507601648953048588754147172089787574344406485317342328659050613470152656721 21

Encrypted B:
111601736525297819827793282449195993684827374257142783985395626779955851912645567403235776496739693470967655461127136094936611558983624968916964071906491 27

Decrypted B: 550

Sign B:
368905794879912707132320011178263870656918773165544758488698214021943963003061696757021252499127284090980721317582699651326888343189180186176169958317833 09

Verify B: True

k =
100816930159492784194464479703985023033215697727829866559416833597200831698454187207420788710857908136930497704870838634374411466716626699789228527867227 90

Key received

Encryption:
817104273926012532834567864723316705779394294761203712004699049242627799352230204291001594342137784750492316509518667947784972445960113033754970398443 2228

Decoding: 550

Checking the text: True

```
Message :  550
-------A-------
Public exponent:  100001
Public n:  193163445556798214071035565830998378119882870795349654125894782207649702712492973646287478800977438515724102992660416055129078751119942090221775328312118 01
Encrypted A:  817104273926012532834567864723316705779394294761203712004699049242627799352230204291001594342137784750492316509518667947784972445960113033754970398443222 8
Decrypted A:  550
Sign A:  191477453448464079136288669792846506074603234839691323277164544477721995206813953055721883698958108176612167761552631951270912115563313932539969186907012 26
Verify A:  True
-------B-------
Public exponent1:  100001
Public n1:  472785812706825727016533937894309437546514708893101940998385640590145305930194507601648953048588754147172089787574344406485317342328659050613470152656721 21
Encrypted B:  111601736525297819827793282449195993684827374257142783985395626779955851912645567403235776496739693470967655461127136094936611558983624968916964071906491 27
Decrypted B:  550
Sign B:  368905794879912707132320011178263870656918773165544758488698214021943963003061696757021252499127284090980721317582699651326888343189180186176169958317833 09
Verify B:  True

k =  100816930159492784194464479703985023033215697727829866559416833597200831698454187207420788710857908136930497704870838634374411466716626699789228527867227 90

Key received

Encryption:  817104273926012532834567864723316705779394294761203712004699049242627799352230204291001594342137784750492316509518667947784972445960113033754970398443222 8
Decoding:  550
Checking the text:  True
```

Перевірка

Get server key

Clear

Key size 256

Get key

Modulus 9FCB31953E17DD39DFE869AB9FDC0B4EBB02073F76CE4C498C473D9CBF5BF1C9

Public exponent 10001

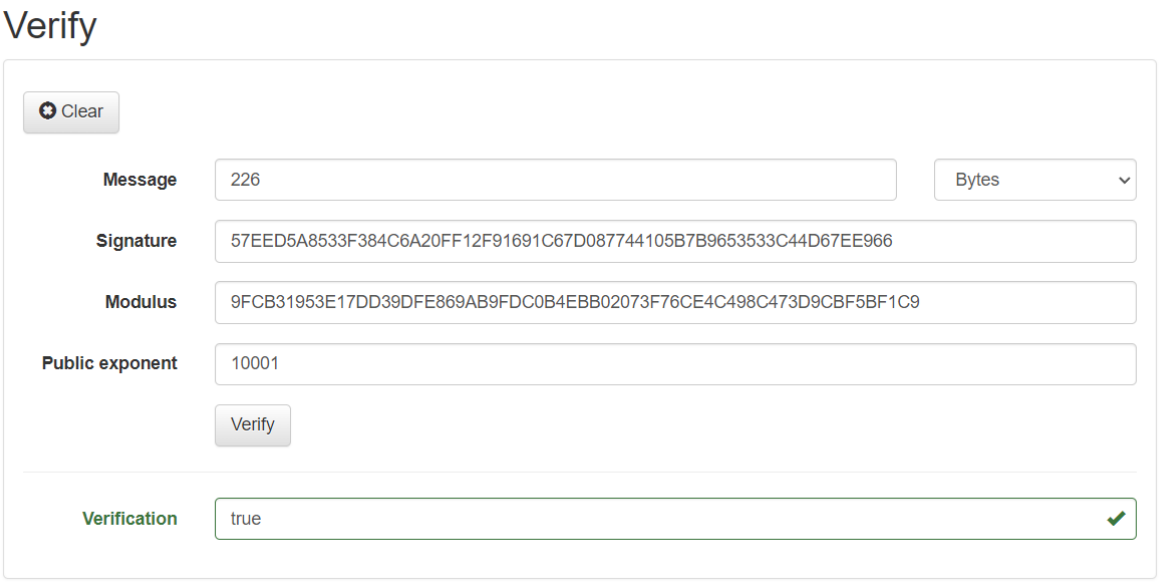
Sign

Clear

Message 226 Bytes

Sign

Signature 57EED5A8533F384C6A20FF12F91691C67D087744105B7B9653533C44D67EE966




```
print('Checking the text: ', 550 == decrypt(encrypt(550, 100001, n), d, n))
print('--------------------')
print('Message : ', 550)
print('Moduls: ', int("9FCB31953E17DD39DFE869AB9FDC0B4EBB02073F76CE4C498C473D9CBF5BF1C9", 16))
print('Public exponent: ', int("10001", 16))
print('sig: ', int("57EED5A8533F384C6A20FF12F91691C67D087744105B7B9653533C44D67EE966",16))
print('ver: ', verify(550, int("57EED5A8533F384C6A20FF12F91691C67D087744105B7B9653533C44D67EE966",16), int("10001",16), int("9FCB31953E17D
```

```
--------------------
Message :  550
Moduls:  722767550880315931445827920463635493234307321448074096861893925898266646901 21
Public exponent:  65537
sig:  39773202038171265103774774595660836612707375241717158745249864045812850288998
ver:  True
```

Висновки

В результаті лабораторної роботи, ми ознайомились з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми. Ознайомились з системою захисту інформації на основі криптосхеми RSA. Вивчення протокол розсилання ключів