МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФІЗИКО- ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра інформаційної безпеки

КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

Криптографія

З теми: « Вивчення криптосистеми RSA та алгоритму електронного підпису; ознайомлення з методами генерації параметрів для асиметричних криптосистем »

Перевірила:

Селюх П.В

Виконали студенти групи ФБ-92

Ханас Максим Любомирович

Гуманков Денис Максимович

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.

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 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.
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.
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().

Код програми знаходиться у файлі **main.py**.

Хід роботи:

Спочатку ми реалізували функцію пошуку простих чисел з заданого інтервалу, після цього за допомогою неї згенерували дві пари p та q. Далі реалізували функцію генерації пар ключів зі знайдених раніше p та q. Далі нами було реалізовано функції шифрування та дешифрування, також було зроблено функцію, яка відповідає за створення цифрового підпису, та ще одну, за допомогою якої цей підпис можна підтвердити. Після цього були створені два абоненти, на яких і було перевірено функціонування програми.

Результати:

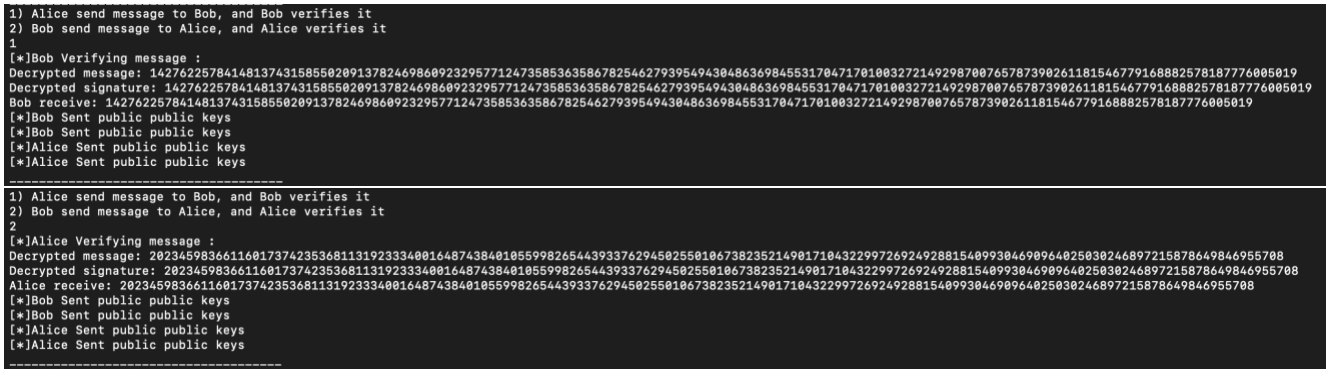
Параметри абонента A:

```
[*]AliceInformation:
e: 279316426292562734591757186309668889941857277161922979633960443550362579504983311488370014025464247387442024830950857582854121797995787791628161629711093 17
n: 306638046776932570684735322594700314242580371365888720944347941330419956839422451101507635192018024060169827655801847599631983517487644221909239419013015 27
_d: 182808230238128093272735554187828499795494725115805690296822957767848858043567493487312840103115640335774509930843081585442140619319394766204792165642145 73
_p: 167196046186767214339522199572021060931943641621845403873058826855452793819571
_q: 183400297896040521923535858048208471468161682483691871773064288873825029319437
_my_message: 1427622578414813743158550209137824698609232957712473585363586782546279395494304863698455317047170100327214929870076578739026118154677916888257818777600501 9
signature: 2898756349608428412291171958654201752939255002801250361506975612844552635856505141423716670547996218045735670675146724597868233472357597509243675693360396 8
encrypted message: 207892826219312004915850908217764334550923030408807864921762536585575042534040891298974880544844200583583668250185485620486881192853535552449413043437755 43
```

Параметри абонента B:

```
[*]BobInformation:
e: 250558832543105334323444847305014692478960325776732212126382322401269278901911210644378161765745079698870856699244383653500084064302796413963707261639221 7
n: 405034178718062235040836032287088434539103651467489697017696196725278482519505270191578492795646126619970484676331465547363209980374670306631624472606661 93
_d: 129387145175659521289382114506527975206812431280359857659436312382489263366065286210897219204539074006188340902560887083647621147012297679377144024770865 45
_p: 201595998273382935234132791017545272731890754275055866329579894380649394227327
_q: 200913798977695080090433214799930515104840167312324613765981729888427453875759
_my_message: 364065370088547569684810863560123240670200928995284536539515789584697198938251252182794162897304739310712863029246803942008563766118368368652849774708469 89
signature: 111489617402898579419548450276610962945152665628483385024142364287589821921309359389555057752252718333591112360293030872008361958962834769080580559465120 82
encrypted message: 286675626713180101592747049413323790983268157600480761864582898022959721178156312998686934555358983874342186461818275610766952374645274639607272315565447 32
```

Приклад Комунікації:



Висновки: під час виконання даної лабораторної роботи ми ознайомилися з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA. Також, ми ознайомилися з криптосистемою RSA та реалізували засекречений зв’язок з використанням цієї системи.