Національний технічний університет України «КПІ» імені Ігоря Сікорського
Фізико-технічний інститут

Комп’ютерний практикум 4
Криптографія

Виконали:
студенти ФБ-21
Князян Кирило Андрійович
Новіцький Олександр Костянтинович

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

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.

Порядок виконання роботи

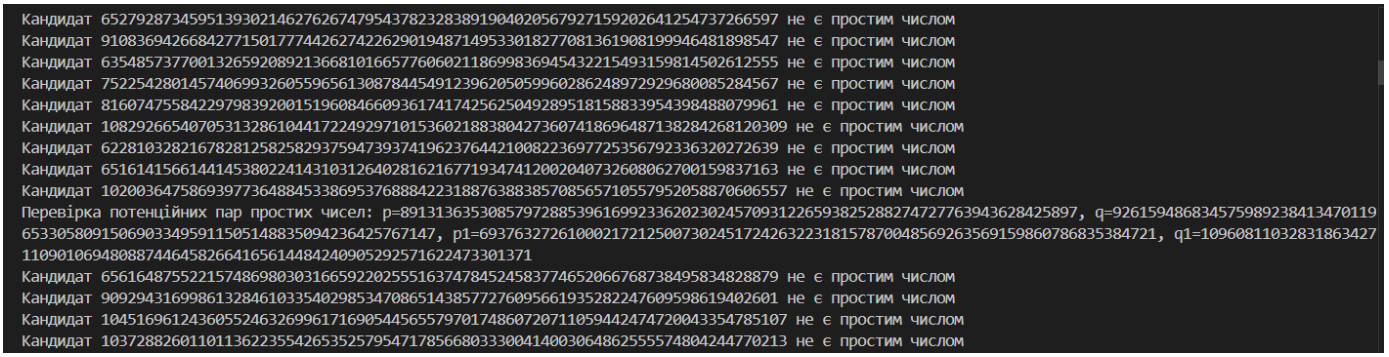
1. Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч, використання готових реалізацій тестів не дозволяється.
2. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p1 , q1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq <= p1q1; p і q – прості числа для побудови ключів абонента A, p1 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), (e1, n1) 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 і B,

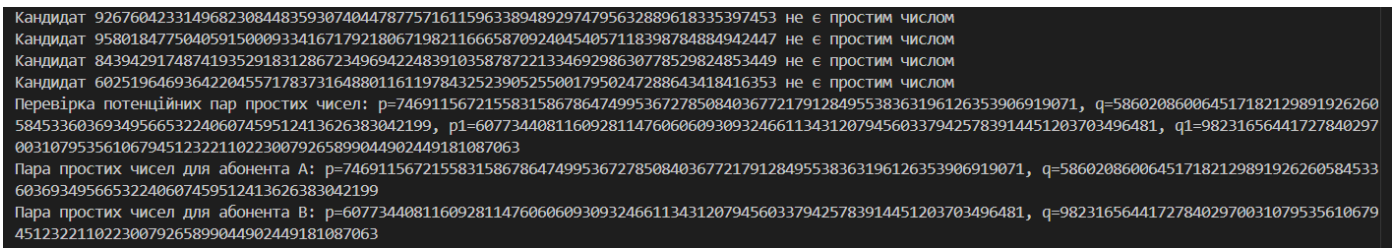
перевірити правильність розшифрування. Скласти для A і B повідомлення з цифровим підписом і перевірити його.

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 $0 < k < n$.

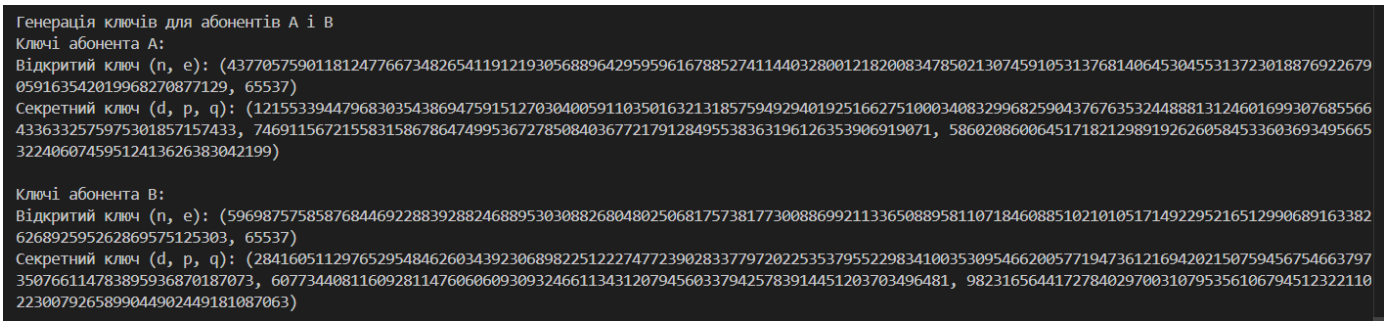
Хід роботи

1. За допомогою вбудованої у python random.getrandbits генеруємо числа довжини 256 бітів (можна і більше), які потім проходять перевірку спочатку діленням на набір простих чисел, а потім тест Міллера-Рабіна, що можна побачити на скриншотах, тобто генерується число заданої довжини біт, якщо воно проходить перевірку, то повертається, якщо ні, тоді виводиться повідомлення, що перевірка не пройдена, потім, коли дві пари простих чисел згенеровано, йде перевірка на умову pq <= p1q1, якщо вона не виконується, тоді генерація йде по новому і так поки не буде отримано дві пари простих чисел, що відповідають умові:

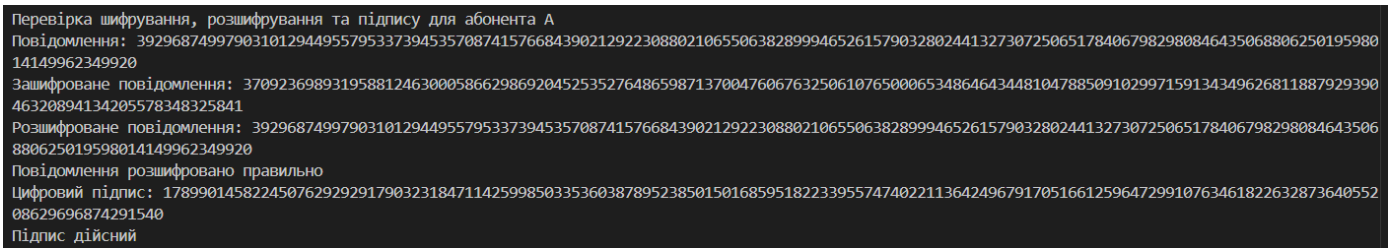


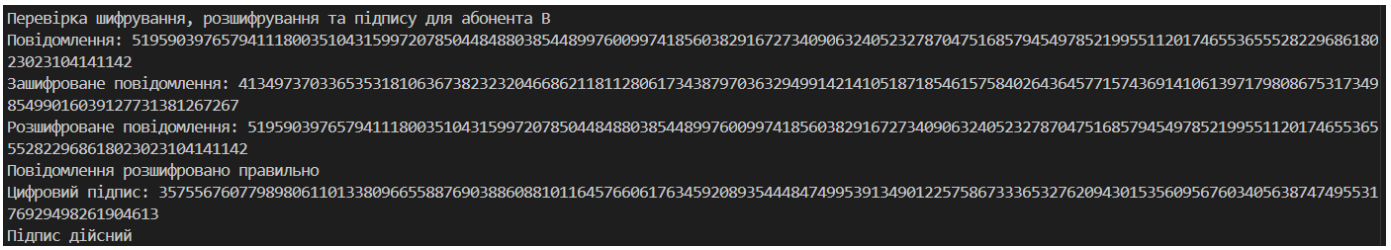


2. Тепер на основі простих чисел генеруємо ключі для RSA, тобто для абонентів A і B на основі їх пар простих чисел обраховується n = p * q, φ(n), e обрали 65537 - ту, яку використовують найчастіше, і відповідно обрахували d = e^(-1)modφ(n). В кінці просто повертаємо відкритий ключ (n, e) і секретний - (d, p, q):



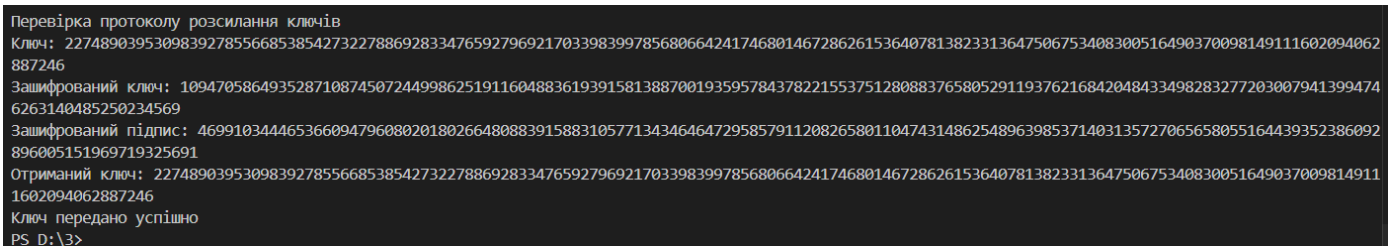
3. Тепер відповідно йде шифрування, розшифрування і створення та перевірка підпису: спочатку генерується повідомлення, потім воно шифрується, розшифровується і згенероване повідомлення порівнюється з розшифрованим для перевірки правильності роботи, після цього генерується цифровий підпис і функція Verify перевіряє його. Можна побачити усі кроки нижче і впевнитись в правильності роботи усіх функцій при роботі з абонентами A і B:





4. Тепер відповідно робота протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA: спочатку генерується ключ 0 < k < n, потім за допомоги SendKey отримуємо шифровані ключ і підпис (відправник A шифрує ключ за допомогою відкритого ключа отримувача B, створює підпис своїм секретним ключем і шифрує цей підпис відкритим ключем отримувача, повертає шифровані ключ і підпис), ReceiveKey розшифровує ключ і підпис і перевіряє підпис,

після успішної перевірки повертає ключ (отримувач B розшифровує ключ та підпис своїм секретним ключем, верифікує підпис, використовуючи відкритий ключ відправника, та в разі успішної верифікації отримує ключ):



Висновки:

У результаті виконання четвертої лабораторної роботи ми детально розібрались у роботі RSA. Ми навчились генерувати та перевіряти числа на простоту за допомоги ділення на прості числа разом із тестом Міллера-Рабіна, отримувати відкритий та секретний ключі на основі пари простих чисел, також шифрувати, розшифровувати та підписувати і перевіряти підпис, а в кінці використали розроблене раніше для реалізації роботи протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA.