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

Варіант 4

Виконали:

Студенти ФБ-01

Новак О. І. Тостоган Є. Г.

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.

Постановка задачі:

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) , (n1, e1) 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().

Хід роботи

В роботі ми використовували функції з бібліотеки random: randint(), randrange(), аби згенерувати число розміром 256 за завданням. Перевірка на простоту, здійснюється в декілька етапів: ділення на відомі перші 13 простих чисел(якщо випадкове число ділилось на них відповідно воно не було простим); тест Міллера-Рабіна(в якому знайшли розклад нашого числа, а потім перевірка простоти за основою випадково вибраного числа x). Реалізована функція test_millrab() повертає True/False значення у відповідності чи є число сильно псевдопростим, чи ні. Для реалізації коректної роботи криптосхеми RSA(обмін повідомленням між абонентами Alice та Bob) ми написали такі функції:
GenerateKeyPair() – генерує пару відкритий/секретний ключ для абонента;
Encrypt(), Decrypt() – відповідно зашифровує та розшифровує обране повідомлення;
Sign() – створюється ЦП з ключем для подальшої автентифікації абонента-відправника отримувачем;
Verify() – перевірка ЦП на коректність;
SendKey() – Шифрування ключа ЦП та самого ЦП відкритим ключем абонента та їх надсилання;
ReceiveKey() – Отримання ключа ЦП та ЦП другим абонентом та його перевірка за доп. функції Verify().

Результати виконання:

---------абонент Alica

Відкритий ключ:

e =
96324235546268314326483608405576254257374742467963393950112086316909531130915624403461491159795299786831535109357432172734332610233954774418233686062364 9
n =
53480623165919955931840229127054931934264554912591482116240714989216447253722420314318682947515049104445151580001628418606608514719264566232765601742653 69

Секретний ключ:

p = 68929859301761385566417125560041225346068272072707753427887310437384778170959
q = 77587019192643774008643184144062837160039466458165551095637367502025299955991
d =
37230472179924218106381581662200281043934972949809968715766668992310968548462508618887586894660525381858518398881612019011682854014848808558489328520380429

---------абонент Bob

Відкритий ключ:

e =
61445771871384520048027296395779448340084040571518663503790902789347252513345948425996162229729281055969495200526956838717458820574483727885087736472987 43
n =
12368354756144646705390593711728566101557437905166210884458301567456210775013795185909178121655653564117429813761018150878922948599785794666927773426118357

Секретний ключ:

p = 11162289745746854251954072141877144010180416512226499546199362874617104279204 3
q = 11080481727198791878542848033392681995739333975234062108357445280473523933279 9

d =
889900125579320368509310294238389951779036931435579569226849362970059105150598976
670281150944283602477149495288962181964538209244907484992876894949889311

```
————————————————————Робота з повідомленням————————————————————
Початковий k: 449545271136583311640544425106599988153137380657753491243917290872554088337815047060256261734123738735
436519471482908374410769511526751063242740283355448 7

Згенероване повідомлення:  342781990337141746764818395236701856698987270891109901184329379026656901926298274429464566
816447707104611083263111888412869105027460564014838955122953318 6

Alice шифрує повідомлення і надсилає Bob
Encoding ...  |##################################| 100/100

Зашифроване повідомлення:  373368297623483847860977555557624113840012024329131111652230807783340146941258066627442920
096391105071081999382633372055684494370019347548598052537641682 8

Bob отримав ключ

Розшифрований k = 449545271136583311640544425106599988153137380657753491243917290872554088337815047060256261734123737
873543651947148290837441076951152675106324274028335544 87

Bob розшифровує повідомлення
Decoding ...  |##################################| 100/100

Розшифроване повідомлення: 342781990337141746764818395236701856698987270891109901184329379026656901926298274429464566
816447707104611083263111888412869105027460564014838955122953318 6

Перевірка тексту:
Processing ...  |##################################| 100/100

Обмін повідомленням між Alice і Bob успішний
```

Перевірка правильності виконання на онлайн ресурсі:

Висновок: Ми ознайомилися з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA. Відпрацювали навички роботи з системою захисту інформації на основі криптосхеми RSA. Дізналися про цифровий підпис і його використання в цій системі.