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

ЛАБОРАТОРНА РОБОТА №4
«Вивчення криптосистеми RSA та алгоритму електронного підпису; ознайомлення з методами генерації параметрів для асиметричних криптосистем»

Виконали:
студентки групи ФБ-04
Андрійчук Анастасія та Стоян Анастасія
Перевірив:
Чорний О.

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 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.

Хід роботи

Для простішої роботи зі зберіганням та використанням змінних ми вирішили використати можливості ООП, реалізовуючи функції, як методи класу. В таких методах необхідні змінні зберігаються в полях класу, а також в деяких методах деякі змінні використовуються без їх вказання в аргументах методу. Зокрема це стосується методів, які використовують приватний ключ.

Далі наведемо результати, що ми отримали. При передачі ключа ми використовували умовний оператор, щоб коректно визначити, хто відправляє.

A:

p = 0xec346f694df50c74fd3a52c8d49e0e9c4759c896343d607efbdd0ab5365dd9e3

q = 0xc5c9a82bacaf8f3bc0b96394706b63d84c3729f2b616b620137efed55a697c8f

public A:
('0xb67e6a1237ecd0b10c8f625c0ca90fbfb34a178e21fc03d95281ca594387aa0b2c488b565dd6fcb15f2735947464534eb623584d598f78cf4c1890860011a9cd', '0x10001')

private A:
0x4eb9da2861d7d76f896b919304232a9818502e39328ff5591b2ec28c1f85b6e4bfccbe82feb80d00a889fea75fd66489b5c052069bb592761cbd77bda0150041

B:

p = 0xcc25813854714b7e4804c1bd6fc1bcee473dfebf49ab628b5397735ae93ac67b

q = 0x8dc2e8ba466db62433c64928e05b3999cdb1485d663d6a65145a56124b971057

public B:
('0x710c162e7bb71270be993c425837e5a241c99bad1a340f0c6294e3c1f74672265950f50535ec0fc75741cbb4e010621f0e64fdeac256592414aef2f00aee23cd', '0x10001')

private B:
0x40ca5598348d9b4304ea1c3046115b4a207c64fdd40bcc282404037932b8ac48ea1d47bfd51fa960946bbe5b65037541651e03349e392ce69cf5196a843e5a1d

msgA = 0x3f3f3f3f3f

crptA =
0x1aa5b11a4987efda16a8eb269090ebd5958619839b78e84efdc91a3c4d686a58c8a807934c1e830424a5e5b949c790733ec887014ccd91fe610256dc6d2522d8

dcrpA = 0x3f3f3f3f3f

msgB = 0xafafafafaf

crptB =
0x33464a6d43ec734bba7a96a00fff4254255d27d7b39aba599036211fbcd7afe612597b201166b05fd273566966dd8057e827411881dd93d4c7fcfffbdf871cd4

dcrpB = 0xafafafafaf

sign A = ('0x3f3f3f3f3f',
'0x8d9c29aa0997186528d25dec058be7df9c35524113a6fa91eb78b66f57da259fc68ca58a727a1134ec7df9d818a5de41217ecd421260737ff2465eb40aaabf08')

Verify: True

sign B = ('0xafafafafaf',
'0x6970b1a08f6aff8f26624a07d50f7d74539b013cf6fa4c19ba4d8431778cdb2fef9b11ea603f82dfd2a2b34328e7fae032706cc1a3ed15f45eb1418332240f51')

Verify: True

k = 0xc00feec00fee

S =
0x2d92e41681a010de424303ad8e9082c1ae1e609559d6e6004b2cf7482b226ad43d2e775048820aae3bc29ed6ec10a069957cae595ce30c667ca2c3aa8f5dda3b

S1 = 0x2d90ed340c85bb5c8d756b4aefc6480e869ea4afdd9ffe14223e4049e07c445d77e016caea6a5a9287514179dcea1ec863b7942603545b37bad2f1ab505ebcb2

k1 = 0x543bd69a9606c09baf24022e3099ef2b9cdc69f1ee0650322e911c5e4eb310d1baf0414d89599a6a0975e701c21d96b8a9b2e6d4fc701f40eafb0decd559f666

k = 0xc00feec00fee

S = 0x2d92e41681a010de424303ad8e9082c1ae1e609559d6e6004b2cf7482b226ad43d2e775048820aae3bc29ed6ec10a069957cae595ce30c667ca2c3aa8f5dda3b

Висновки:

Ми ознайомилися з імовірнісними тестами для визначення псевдопростоти, з генеруванням криптосистем RSA, шифрування повідомлень, цифровими підписами, передачею ключів.