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»

Виконав:

студенти групи ФБ-04

Ляденко Максим

Петриковець Віталій

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 і рекомендації щодо 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 $pq \leq p1q1$; 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 $0 < k < n$.

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().

Кожну операцію рекомендується перевіряти шляхом взаємодії із тестовим середовищем, розташованим за адресою http://asymcryptwebservice.appspot.com/?section=rsa.

Наприклад, для перевірки коректності операції шифрування необхідно а) зашифрувати власною реалізацією повідомлення для серверу та розшифрувати його на сервері, б) зашифрувати на сервері повідомлення для вашої реалізації та розшифрувати його локально.

Хід роботи

В ході роботи ми написали функцію для пошуку простого числа, використовуючи для перевірки алгоритм Міллера-Рабіна. Реалізували функції для генерації пар ключів, шифрування та дешифрування повідомлень, функції підпису та верифікації повідомлення, відправки та отримання ключів. Найбільше проблем було із розумінням алгоритму Міллера-Рабіна.

Результат роботи

2. Ми зробили так, щоб першому автоматично давало менший добуток.

p = 0x536306c0af60008cb868591aa0aa6adc22fb153cbb4a3f02dc865195aff71d27
q = 0x70a24d43bb199af3ca4588b5eef2cd3ef0979eed9d70176d8fdd91b090ec26b3
p1 = 0xfe08d64ec832b37778dcebff20ca4b5b23118264bc06782364644bf3fcfbcf07
q1 = 0x60e715ad3035f33fdc1185d1cb9940e73507c2922932292d090cc71f74be555f

3.

n = 0x24b030c96b8d613f13e32e097ae84620b58551b5a0fc725691786299487f76e3ae4167aaf111a2ae6159f12c101fa433ea22693234eb113f70129dcd88112c45
e = 0x10001
d = 0x39f5e01d55bcd5954165f6a5237730d31bf2b353ae3124b17ec02184e7f3f76e257f23cc41768c60fb17f86b266ae53b983d10528a9ab185badb9ff377901f5
n1 = 0x60289fd9818e241b16f0719d1541b78fa3108154f2919efb0fd58ee875f8defb434616506a32a5ffe4e23d63a7c49a68c1c5267bbd5a5fd5cfe8921650612699
e1 = 0x10001

d1 =
0x174d80ff6ae0116be73139b741917d7477a5e651b9bf0471ff3257c189e557d3e33be6623479293e6274485f34bd9e7a3c9d84b776d126a4986a31037705ae09

4.

message A = 0x42434445

message B = 0x56575859

encrypted A =
0x1a6ee22785264dd31b0832e0fca2615922366aab569987a8cc3d12b2f470db512aa9c2943f7bcd8fb1a4bbee86e44d91c0f12b15adfc8166a72a1bf89e4306a9

encrypted B =
0x2169e430522c27ea84ed2faa6de33927655821ea031e143fa38e2377914c7571ebbb0281ccebbf2a632e7c765e25b206cbf26936cc617af59e1142efd0b99b7b

decrypted A = 0x42434445

decrypted B = 0x56575859

sign A = ('0x42434445',
'0x2444659d5b5cf6a48aa103e137b617601a8bcc36e0e56699a2532e0c0adc1387e331ee0870a59162cb3b85c2b8c9b56378d77cf9030ed0033c12d0607678ad11')

sign B = ('0x56575859',
'0x2788eb055ffcdaafdcf7687c0d722b1a9d709ba554accd4e439bfefa943bf01f8b6ceadfd9f86048064e534d883daf7842f5ea01dd1ec6be410722fbb87b147a')

is message A verified? - True

is message B verified? - True

5.

S =
0x1e8e7270eae9a982e8bbc93f06b1b640619f6e6d51c535a76d2d41616c45a01a84963c8cbcb7c0168a2757a79384a05c3af556ba03e9711ec628a8fe4a48349b

S1 =
0x303a401fcd6b1c8f59ee42750bf53fc457bbd2bebac02ab9deb81847c45c1ffeff29ae51795045ce48c6b3373255260464a7ef7f98fe2e358b1a1774c6ce9e00

k1 =
0x4e46681166fb7c5b1a8ede6ac71b5c961fcaf331aa6fe34d03feccf05258733f09c73413ac91aa1f63419904a306a871b2bb70a716e081061e948ee185339a28

k = 0x11

S =
0x1e8e7270eae9a982e8bbc93f06b1b640619f6e6d51c535a76d2d41616c45a01a84963c8cbcb7c0168a2757a79384a05c3af556ba03e9711ec628a8fe4a48349b

is the key not deformed? - True

Висновки:

За роботою ми освоїли загальні принципи роботи RSA, використали алгоритм Міллера-Рабіна для визначення простих чисел. Також, нами був обміркований принцип шифрування та дешифрування різними сторонами обміну повідомленнями з відкритими та таємними ключами.