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

Виконав:
студент ІІІ курсу ФТІ
групи ФБ-95
Чорний Анатолій
Перевірила:
Селюх П. В.

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 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 A, 1 p 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) , (,) 1 n1 e 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().

Хід роботи

Bob	
n	**77700524685834816716896983924135766340300518174817986800006541046983538516320930774493667041715552326272293525561633911748170015662050787492460210041264496** **59884569011406365266728196215420387268492459418711174236039818671808314017188675069926994758419053978148469659029926660554636757973566003933294937349456990** **68630878199665197024311420490169701366623233588155444456180876208022187753536924737413021980819440139536631270519086043315469042211804974747405457691895790**
e	**49574555052117715311902475848469229293031401399170965932928929063256174359678617179948972017769127781221107328223176216432335541746775416938941909111951590**
d	**97456240659043140797316416497791664815199528965710678720283330866027909378426677070010371067569435365297901106205066153224895988277074289287140784469279**

```
Alice = abonent()
Bob = abonent()

msg = 'hello, my name is tolya its pleasure for me to meet you'

Bob.GenerateKeyPairReceiver()
Alice.GenerateKeyPairSender(Bob.n)

packet = Alice.SendKey(msg, Bob.e, Bob.n)
output = Bob.ReceiveKey(packet, Alice.e, Alice.n)
```

```
C:\Users\morning star\Desktop\backup\third\crypto\4>python mr.py
hello, my name is tolya its pleasure for me to meet you
```

1. В якості теста на простоту числа було обрано ймовірнисний тест Міллера-Рабіна. Також мали місце спроби реалізувати тести Соловея-Штрассена та Ферма, але за низької швидкодії їх було відкинуто.
2. Була написана функція вибору випадкового простого числа getran().
   2.1. За допомогою генератора псевдовипадкових чисел вбудованного у мову Python, було обрано випадкове число з проміжку від найменшого числа довжиною X біт до найбільшого числа довжиною X біт.
   2.2. Потім базуючись на постулаті Бертана було побудовано алгоритм що на основі випадкового числа перевіряв методом Міллера-Рабіна, кожне наступне непарне число.
3. Для створення ключів була створена функція genkey() що створює пару різних ймовірно простих чисел, та на їх основі вираховує значеня e, n, d (відкритий та таємний ключ)
4. Щоб привести умови задачі до умов реального світу, були написані функції encode() та decode() що перетворють строкове значення у числове, для того щоб повідомлення могло бути обробленно у нашій криптосистемі.
5. На основі формул представлених у методичних матеріалах були написані функції: encrypt(), decrypt(), sign() та verify(). Для демонстрування коректного обміну ключами було створено класс Abonent що має методи реалізовані на основі функцій що були написані раніше (Encrypt(), Decrypt(), Sign(), Verify()). Для демонстрації було створено 2 об'єкти классу Abonent: Alice і Bob – класичні герої криптографічних задач.
6. Через вимогу для числа n (n відправника повинен бути менше за n отримувача), були створені 2 різних функції для створення ключа для об'єкту классу- GenerateKeyPairSender(n) та GenerateKeyPairReceiver().
7. На основі вже написаних функцій, були написані функції
   7.1. SendKey() що приймає на вхід повідомлення та відкритий ключ отримувача, та віддає на вихід пакет що містить у собі зашифровані підпис та повідомлення.
   7.2. ReceiveKey() приймає на вхід пакет що був відправленний раніше, та відкритий ключ відправника, розшифровує підпис та повідомлення. Якщо перевірка розшифрованного підпису не дає позитивного результату повідомлення не розшифровується. Інакше ми розшифровуємо і декодуємо повідомлення.

Висновки: в ході виконання компьютерного практикуму мною був дослідженний метод побудови криптосистеми RSA. Для досягнення результату були реалізовані ймовірнісні тести на простоту чисел, реалізовані методи вибору ймовірно простих чисел у проміжку. Були вдосконаленні знання щодо організації засекреченого зв'язку й електронного підпису за допомгою RSA, були проаналізванні протоколи розсилання ключів.