МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФІЗИКО- ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра інформаційної безпеки

Лабораторна робота №4 з дисципліни Криптографія

з теми:

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

Виконала:

Бородай Ю.

Групи ФБ-96

Мета та основні завдання 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 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq \leq 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 е та секретні d і d1.
- 4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів A 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 A и B, перевірити правильність розшифрування. Скласти для A і В повідомлення з цифровим підписом і перевірити його.
- 5.~3а 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 Епстурt(), 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

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.

Тест простоті Міллера-Рабіна:

```
def millerrabin_test(n):
for prime in [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41]:
  if n % prime == 0:
     return False
if n == 3 or n == 2: # числа прості
  return True
if n < 2 or n \% 2 == 0: # парні тому не прості
  return False
else:
  s = 0 # встановлюемо лічильник
  d = n - 1 \# для розкладу
  while d % 2 == 0: # пока d \ на 2, увеличиваем счетчик
    s += 1
     d = d // 2
  for i in range(6):
     a = random.randrange(2, n - 2)
     x = pow(a, d, n)
     if x == 1 or x == n - 1:
       continue
     for i in range(1, s):
       x = pow(x, 2, n)
       if x == 1:
         return False
       if x == n - 1:
         break
     if x != n - 1:
```

return False

return True

Частина варіантів ключів не пройшовших тест на простоту:

157679868220163380271067298285262327966643469870872099528739683673083882050507 Is not a prime number.

157679868220163380271067298285262327966643469870872099528739683673083882050509 Is not a prime number.

157679868220163380271067298285262327966643469870872099528739683673083882050511 Is not a prime number.

 $157679868220163380271067298285262327966643469870872099528739683673083882050513 \ \ Is \ not \ a \ prime number.$

 $157679868220163380271067298285262327966643469870872099528739683673083882050515 \ Is \ not \ a \ prime \ number.$

157679868220163380271067298285262327966643469870872099528739683673083882050517 Is not a prime number.

 $157679868220163380271067298285262327966643469870872099528739683673083882050519 \ Is \ not \ a \ prime \ number.$

 $157679868220163380271067298285262327966643469870872099528739683673083882050521 \ \ Is \ not \ a \ prime number.$

 $157679868220163380271067298285262327966643469870872099528739683673083882050523 \ \ Is \ not \ a \ prime number.$

 $157679868220163380271067298285262327966643469870872099528739683673083882050525 \ \ Is \ not \ a \ prime number.$

157679868220163380271067298285262327966643469870872099528739683673083882050527 Is not a prime number.

 $157679868220163380271067298285262327966643469870872099528739683673083882050529 \ Is \ not \ a \ prime \ number.$

• • • •

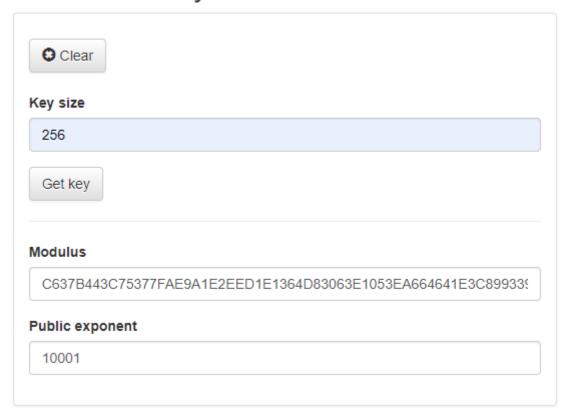
157679868220163380271067298285262327966643469870872099528739683673083882051079 prime

121109653385964961222211875386845537491656264703991151153008036834481440668375 Is not a prime number.

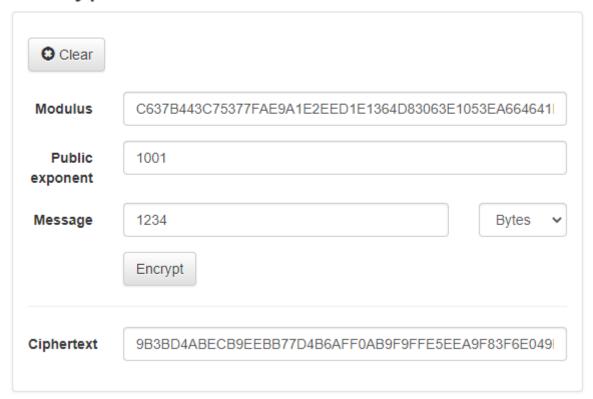
• •

121109653385964961222211875386845537491656264703991151153008036834481440668699 prime

Get server key



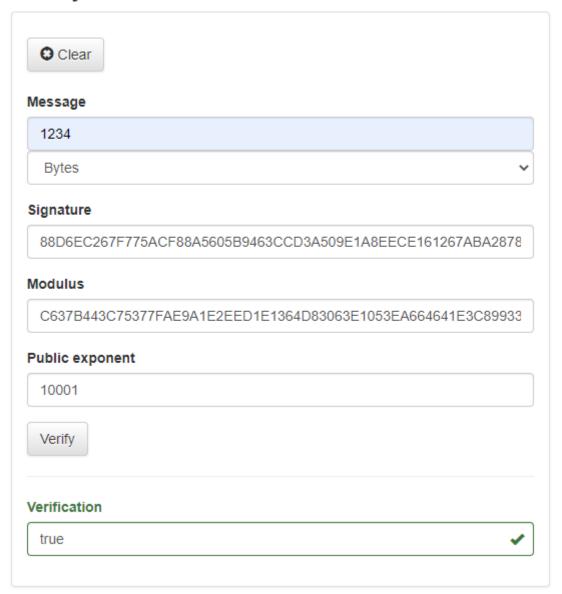
Encryption



Sign



Verify



Висновок

Ознайомилась з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA; практично ознайомилась з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчила протокол розсилання ключів.