

Міністерство освіти і науки України Національний технічний
університет України "Київський політехнічний інститут імені
Ігоря Сікорського"

Фізико-технічний інститут

КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

Виконав: Маслюк В.О. ФБ-25

Київ 2024

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.

Хід 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 p_1, q_1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $pq \nmid p_1q_1$; p і q – прості числа для побудови ключів абонента А, p_1 і q_1 – абонента В.

Перевірка:

```
print(miller_rabin_test(7))
print(miller_rabin_test(10))
print(generate_prime_pair(256))
```

```
a = generate_prime_pair(256)
for i in a:
    print(i)
```

```
True
False
105126369585780747889117880311015803794117943355871915848105204284858108154283
80814331234829204505183036521717853929013580541449192486606709489393475599233
73301290402714979432872716254409627398514506144113351370461118155524538557631
73159555554228835361309872293408161505615986828897924202395053501814048303477
104130249885491664045800963685671483389587097696555116473512251663791189964311
```

Функція перевірки визначає, чи є число простим.

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) , (e_1, n_1) та секретні d і d_1 .

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 і 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 $0 < k < n$.

```
Keyid_A:
Public_A (e, n): (65537, 5728203297400613359952103277346677943355294562862091526355276918465721842663872666382011608518053152552164605383037864842578056160692815898037747229983087)
Secret_A (d, n): (3265942054269669328897726338729816256945144523489096417276641856297015157437364407656705343973793935569879363481369082430375259732953433786726095274581873, 5728203297400613359952103277346677943355294562862091526355276918465721842663872666382011608518053152552164605383037864842578056160692815898037747229983087)

Keyid_B:
Public_B (e, n): (65537, 7317474757772040285152554558357212874009859683323452014427698931316578803541159169202235884760441363591383233697551721149584517233534306150759960304692229)
Secret_B (d, n): (26774653202217606243489671225324010058250520348214959382635186287588928347179886273667647907761124763533183988724367916201722156284376594757169360805633, 7317474757772040285152554558357212874009859683323452014427698931316578803541159169202235884760441363591383233697551721349584517233534306150759960304692229)
```

Шифрування:

```
Шифрування
Повідомлення для B:
363
Зашифрований текст від A для B:
1495366559122381983966863763487686796254469283577672505663848552472023714610316639447783033954225982487839481986223152658112694922553344979658720757252620

Повідомлення для A:
222
Зашифрований текст від B для A:
1771437772991886006047624588200557994699394939192342735276849520282801042255008323013703831570661122678937421385774195260510301330134346431850142950989802
```

Розшифрування:

```
Розшифрування
Розшифрований текст A від B:
222
Оригінальне повідомлення від B:
222

Розшифрований текст B від A:
363
Оригінальне повідомлення від A:
363
```

Обмін ключами:

```
Обмін ключами
А генерує k1 та S1
k1: 3034797386024111638996757271828738737050289114258190062252987709183788487961851634012952061291737791359036405367990928300856516639868276945775310150106870
S1: 712344076180648296875705674823350929837866693487494123365058515616666550325196737121900296859982078748104147594083254951139298529331981018893589288977661
А надсилає повідомлення (k1, S1) до В. В отримав повідомлення
В перевіряє підпис
Перевірено? True

В генерує k2 та S2
k2: 5315294053966389434204690938207410930271510152919838481094369213007214362073694202704018998986546911690579709336864315220103606349501633600638880982438707
S2: 1631380132565712771375592424311096794138980519336036595424394710492980137146481153432005630700465723372869772286972822428031114104263180025419979321524781
В надсилає повідомлення (k2, S2) до А. А отримав повідомлення
А перевіряє підпис
Перевірено? True
```

Згідно з методичкою, розмір ключа повинен бути як мінімум 256 біт. У коді ми генеруємо прості числа p та q розміром 256 біт кожне, що дає модуль n розміром близько 511-512 біт. Тому в полі Key size на сервері слід вводити значення 512.

Get server key

Key size

512

Modulus

A605EEAF3067C424B0A08296DD4A2516F1466A009C2FDBE007

Public exponent


10001

Перевірка через сервер

```
Public_A:
e: 10001
n: 79817D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9CD0CB035B9894E7CF4F621E2259E14058BC26B490BAA165C50
79785EFDB308480B08C8AA84F7AA1

Secret_A:
d: 249032875075529161820627686930115194154017438146934622162242869351129536975182328013892507592593632
809670362807849143002678896382805883788265703186484637
Введіть експоненту сервера: 10001
Введіть модуль сервера: A605EEAF3067C424B0A08296DD4A2516F1466A009C2FDBE00721145DB6A1E7FE52C8AEC6500A2D0
C7CB771C347753578FF864F44530FB15BA624AC2836520F
Зашифроване повідомлення від сервера: █
```

Encryption

 Clear

Modulus

Public exponent

Message Bytes ▼

Encrypt

Ciphertext

```
Public_A:
e: 10001
n: 79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9C00CB035B9894E7CF4F621E2259E14058BC26B490BAA165C5C3
79785EFDB308480B08C8AA84F7AA1

Secret_A:
d: 2490328750755291618206276869301151941540174381469346221622428693511295369751823280138925075925936329
809670362807849143002678896382805883788265703186484637
Введіть експоненту сервера: 10001
Введіть модуль сервера: A605EEAF3067C424B0A08296DD4A2516F1466A009C2FDBE00721145DB6A1E7FE52C8AEC6500A2D6E
C7CB771C3477753578FF864F44530FB15BA624AC2836520F
Зашифроване повідомлення від сервера: 1A7A466BCAA94FBACDDCBDD42DBB7C20BF8C7C907EFB3627379278ADF2B56380A7
DF62599EC8A7BC1B9AE602B8A63BA395456AA5F61070FA86C67AFE985CA09F
Розшифроване повідомлення A: 67234

Повідомлення для сервера:
E3
Зашифрований текст від A для сервера:
95E57615E3B11EBFA762F9C1E79B6B08CFF88E898BE8A1D0AF39235E9B8868269D5A34A59A181268572CE007EDDD6CFED82BF49
D8B7B0136EABEE9FF22363834

Використання функції перевірки
Повідомлення від сервера: 
```

RSA Testing Environment

Server
Key

Encryption

Decryption

Signature

Verification

Send
Key

Receive
Key

Decryption

✖ Clear

Ciphertext

95E57615E3B11EBFA762F9C1E79B6B08CFF8

Bytes



Decrypt

Message

E3

RSA Testing Environment

Server
Key

Encryption

Decryption

Signature

Verification

Send
Key

Receive
Key

Sign

✖ Clear

Message

00009

Bytes



Sign

Signature

2623B6FBF0E2D9581F9F86D7A86C97B6F4D3DF104775BB02C0

```
Використання функції перевірки
Повідомлення від сервера: 00009
Підпис від сервера: 2623B6FBF0E2D9581F9F86D7A86C97B6F4D3DF104775BB02C029A87B3308889947EDCEBC6A012261DF2E
10347DCB470B90C6B3C2E72DD2610EA73601855F8C80
Перевірка? True

Використання функції підпису
k: 110
SA: 8F8CE9B883A030954D03EF77CC2AF75DBDA40B9B2F978B9EF467C1823DD0678ED937DAC417B9537256EA21C422E43D3BF8F
99F9940F8175A664EC92962F5875
Модуль: 79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9C
C5C379785EFDB308480B08C8AA84F7AA1
Показник: 10001

Сервер надсилає ключ, і ми намагаємося його перевірити
к від сервера:
```

RSA Testing Environment

Server
Key

Encryption

Decryption

Signature

Verification

Send
Key

Receive
Key

Verify

✖ Clear

Message

110

Bytes



Signature

8F8CE9B883A030954D03EF77CC2AF75DBDA40B9B2F978B9EF

Modulus

79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9C

Public
exponent

10001

Verify

Verification

true



RSA Testing Environment

Server
Key

Encryption

Decryption

Signature

Verification

Send
Key

Receive
Key

Send key

Clear

Modulus

79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9C

Public
exponent

10001

Send

Key

53BD7D2D3D00078ABDD5708814B4689EFB1D5ACE6CEA2DD6

Signature

58705000032D4A6DB35AA7DC9F7F629EBB460B1FE2A2A716E2

```
Сервер надсилає ключ, і ми намагаємося його перевірити
k від сервера: 53BD7D2D3D00078ABDD5708814B4689EFB1D5ACE6CEA2DD646A092A4D43EA11A25175978B9DE34451A609EBD5
8C05E1CDFC0B8DB1B3177D96A3347C3732BF82A
S від сервера: 58705000032D4A6DB35AA7DC9F7F629EBB460B1FE2A2A716E2F400EF1C7CE4F28363572129030B8DC8A16D906
910350818BA0A09DBC7F07F25D97DAE55A8AC5B
A перевіряє підпис сервера
Чи було його перевірено? True
```

Відправка ключа та S для сервера

Нові ключі A:

Public_A (e, n):

e: 10001

n: 79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9CD0CBD35B9894E7CF4F621E2259E14058BC26B490BAA165C5C379785EFDB308480B08C8AA84F7AA1

A генерує K1 і S1

K1: 72D6EBD7B0E0708E8C37DE55CAABD0BDF209E131604570BC25A7FA6889FBD8B90EF03659341DE53547088BC0F49D2373556731212CA01EA905FF6E7D643345802

S1: C25CC980829F4CCC863C32751CBFC78D83B6302238773A44442A6900FAA48A920D8B29B4AA2EC1784CC8F8F3A40959A1764636E7E4613A23D4355A022749373

Модуль: 79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9CD0CBD35B9894E7CF4F621E2259E14058BC26B490BAA165C5C379785EFDB308480B08C8AA84F7AA1

Показник: 10001

Ключ (для перевірки): 207

RSA Testing Environment

Server Key

Encryption

Decryption

Signature

Verification

Send Key

Receive Key

Receive key

✖ Clear

Key

72D6EBD7B0E0708E8C37DE55CAABD0BDF209E13160457DBC!

Signature

C25CC980829F4CCC863C32751CBFC78D83B6302238773A4444

Modulus

79B17D213E361BE64D8BEEFAD8CD38C0B5A02980C00F82A9C

Public exponent

10001

Receive

Key

0207

Verification

true

✓

Висновок: Під час виконання лабораторної роботи я ознайомився з тестами перевірки чисел на простоту та методами генерації ключів для асиметричної криптосистеми RSA. Я також практично вивчив систему захисту інформації на основі криптосхеми RSA, організував схеми засекреченого зв'язку та електронного підпису з використанням цієї системи, вивчив протокол розсилки ключів та зміцнив навички, отримані у попередніх лабораторних роботах.