Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут ім. Ігоря Сікорського”
Фізико-технічний інститут

Лабораторна робота № 4

з предмету «Криптографія»

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

Варіант 3

Виконали:
Студентки ФБ-33
Яремко Аліна,
Журавльова Марія

Київ - 2025

Мета: 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.

Постановка задачі:

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, 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) , (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.

Виконання:

Отриманні значення для користувача A:

```
==================================================
 1. AUTOMATIC PROTOCOL TEST
==================================================
 > Generating keys for User A and B...
 --- Gemerating 256-bit primes ---

 --- RSA PARAMETERS ---
 p = 28373702912552741737939573701637106296 7
 q = 31336754722391110099189326504962448267 7
 n = 88913976873665952449092140791456703812527455764592258286154770734609967722659
 phi(n) = 88913976873665952449092140791456703811930351188242819767783481732543972177016
 e = 65537
 d = 64958744109604129015098825107877183553034548650274901360780522534662944410 57
```

Для користувача B:

```
 --- Gemerating 260-bit primes ---

 --- RSA PARAMETERS ---
 p = 732285556686703529235863848409438316607
 q = 125125127178570614102608044720411526191 1
 n = 91627323411454159863000520987175601038149575105466696901546698608763674585597 7
 phi(n) = 91627323411454159863000520987175601037951221422619455934520504179202319227746 0
 e = 65537
 d = 11790182924589055497271516753663607482079476482856247187021246192886661855861 3
Done.
```

Шифрування\Дешифрування: повідомлення (message) було зашифровано публічним ключем B і успішно розшифровано приватним ключем B.

```
 === ENCRYPTION / DECRYPTION TEST ===
- Messege: 3267274731941266232618513463401409068125052756156089479026480198433268223277 1
- Ciphertext for B: 7461630665043824209229992553013752265666053861944193709662795088969831418627 64
- Decrypted by B: 3267274731941266232618513463401409068125052756156089479026480198433268223277 1
```

Цифровий підпис: повідомлення було підписано приватним ключем A, а потім успішно перевірено публічним ключем A.

```
 === DIGITAL SIGNATURE TEST ===
- Signature: 8125760765932397271819960129455594940269923335895689175203758268995076173715 3
- Verify: True
```

Протокол обміну ключами: згенерований ключ k є ідентичним ключу, розшифрованому одержувачем B.

```
 === KEY DISTRIBUTION PROTOCOL (A -> B) ===
- Secret key k chosen by A = 385288169371667969224713380861076303110188614314451053085979073685200704971 45
- Encrypted k sent to B = 113511575214502406723548628727178515629318656259744031837718476618626648885148
- Signed k sent to B = 609070007450074712379299134205316946454837668222328143889245508579615980011906
- Received k by B = 38528816937166796922471338086107630311018861431445105308597907368520070497145
- Signature valid? = True
 [RESULT] SUCCESS: Keys are IDENTICAL and Authenticated.
```

Далі перевіряємо функції за допомогою сайта http://asymcryptwebservice.appspot.com/?section=rsa. Для цього було реалізовано програму (web.py), яка надає можливість виконувати шифрування, підпис, перевірку підпису та обмін ключами через HTTP-запити.

Отримання публічний ключів:

```
======================================================================
  2. WEB TESTS WITH RSA SERVICE
======================================================================

[LOCAL]:
 n = 407057126409191461001457894933440449218749059179019767371422405741940750678 89 (hex = 59fe9decf5475cd674498fc4d09e3a04a8f53e3620f220fb776d6d24d01065f1)
 e = 65537 (hex = 10001)

[SERVER]:
 N = A237528B210A202AD232C5ED01B7D7A71AC2ADB34FD7660220B467D474908B55
 E = 10001
```

Перевірка конфіденційності (TEST 1, TEST 2)

```
Messange: 15219350176160349566 (hex = d335fe598250fd7e)

TEST 1: Local Encrypt -> Server Decrypt
Cipher (hex): bdb191c90371a9a8ffc68b30b7deb931ba1c98f0e4b3dd31e8d4148d5c5fb83
Server decrypted: 15219350176160349566
Result: OK

TEST 2: Server Encrypt -> Local Decrypt
Cipher from server: 317338ffeb32627e9da6cdf09b2d62e40527c6b876ea9a5616f8a159e4db1118
Local decrypted: 15219350176160349566
Result: OK
```

Перевірка автентифікації (TEST 3, TEST 4)

```
 TEST 3: Local Sign -> Server Verify
 Local signature: 33016c2672ef2218afae57881518f545faa1d95f970281aec4b44f5a99302daa
 Verification: True
 Result: OK

 TEST 4: Server Sign -> Local Verify
 Server signature: 10620cde35690fbbd0aaae25bffb1700b8a8022c1e4dbd2d205069119793ad59
 Local verification: True
 Result: OK
```

Перевірка протоколу обміну ключами (TEST 5, TEST6)

```
TEST 5: Client -> Server Key Exchange
Key sent = 4251992041277877582 (hex 3b021c3bc827ed4e)
Server received key: 4251992041277877582
Signature valid: True
Result: OK

TEST 6: Server -> Client Key Exchange
Key received: 6718913464402033018
Signature valid: True
Result: OK

======================================================================
FINAL RESULT: WEB PROTOCOL TESTING COMPLETED!
======================================================================
```

Висновки:

У ході виконання лабораторної роботи ми успішно реалізували алгоритм RSA, включаючи базові криптографічні операції: генерацію ключів, шифрування, дешифрування, створення цифрового підпису, перевірка підпису. Для перевірки коректності реалізації було проведено інтеграцію з веб-сервісом http://asymcryptwebservice.appspot.com/?section=rsa. У результаті всі тести завершились успішно, що підтвердило, що локальна реалізація відповідає стандарту RSA та коректно виконує операції над великими числами.