Міністерство освіти і науки України

Національний технічний університет України

"Київський політехнічний інститут імені Ігоря Сікорського"

Фізико-технічний інститут

КРИПТОГРАФІЯ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Виконала:
студентка гр. ФБ-24 Тішевська Анна

Київ–2025

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

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.

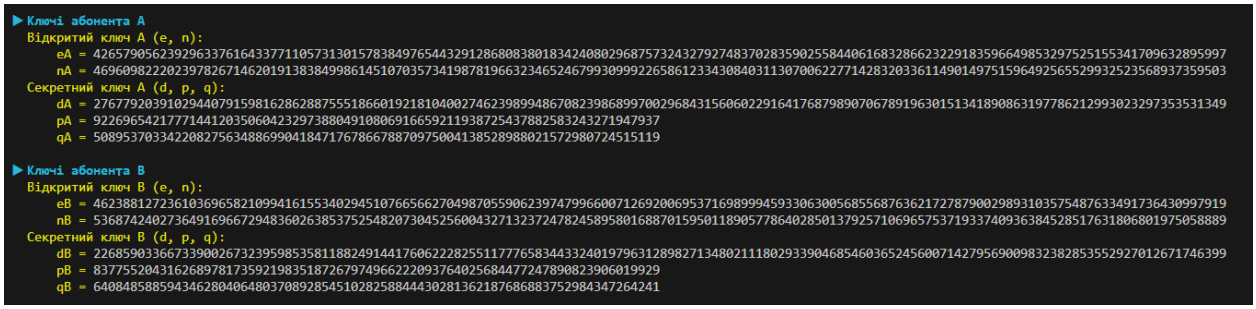
Порядок і рекомендації щодо виконання роботи

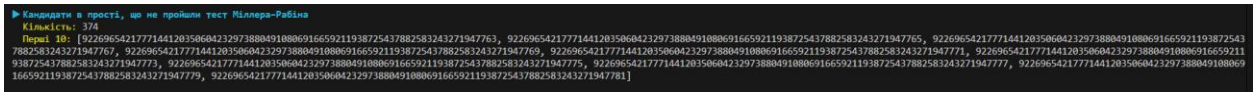
1. Написати функцію пошуку випадкового простого числа з заданого інтервалу або заданої довжини, використовуючи датчик випадкових чисел та тести перевірки на простоту. В якості датчика випадкових чисел використовуйте вбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту 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 ≤ 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().

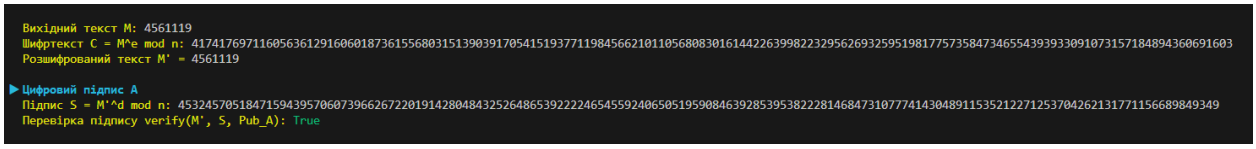
Результати роботи:

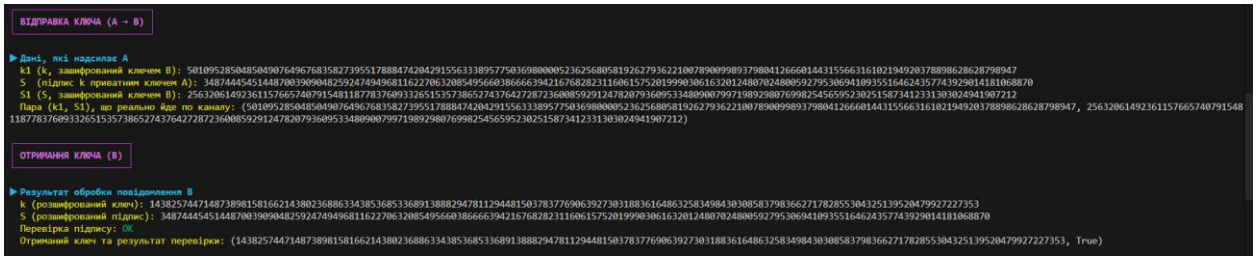
Програма виконала генерацію двох пар ключів RSA (для абонента A та абонента B), використовуючи прості числа p і q довжиною щонайменше 256 біт[1].





Кількість кандидатів у прості числа, які не пройшли тест Міллера–Рабіна: 374. Це свідчить про те, що генератору довелося виконати 374 ітерації перевірки, перш ніж знайти чотири прості числа.



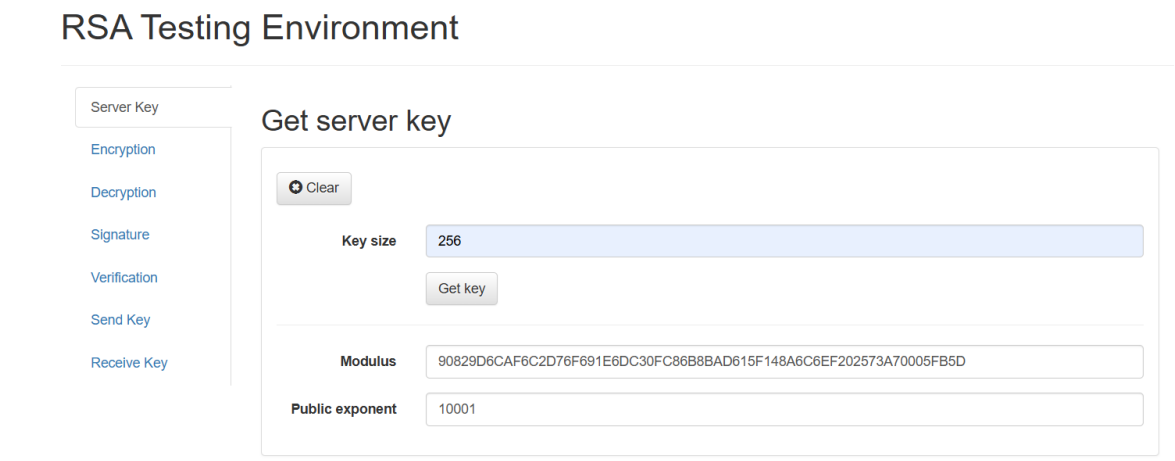


Для процедури розкриття ключів спочатку генерується значення k, де $0 < k < n$. Далі формується повідомлення у вигляді пари (k_1, S_1), де
$k_1 = k^e \, mod n$,
$S_1 = S^e \, mod n$,
а $S = k^d \, mod n$.

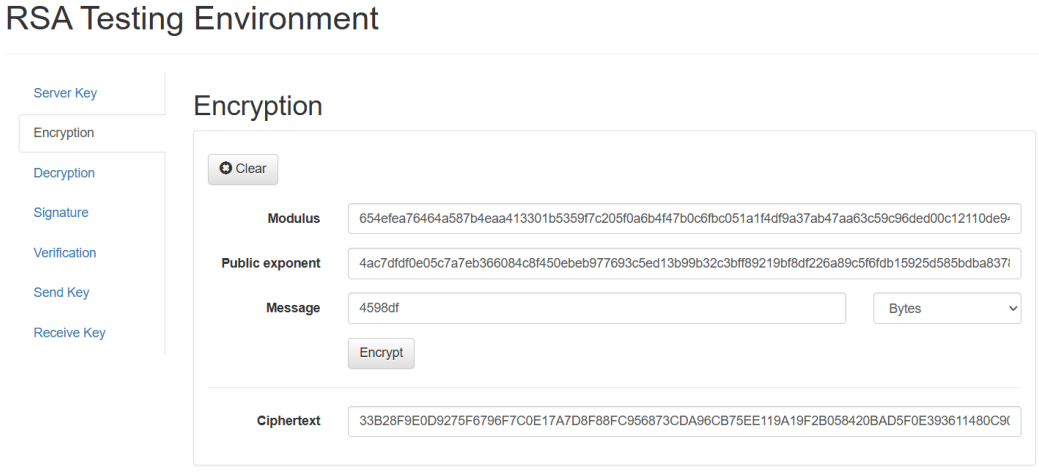
Після цього абонент B отримує повідомлення і виконує такі кроки:
$k = k_1^d \, mod n$,
$S = S_1^d \, mod n$,
після чого перевіряє підпис за умовою $k = S^e \, mod n$.

Результат: перевіряється коректність підпису та відновлюється вихідне значення ключа.

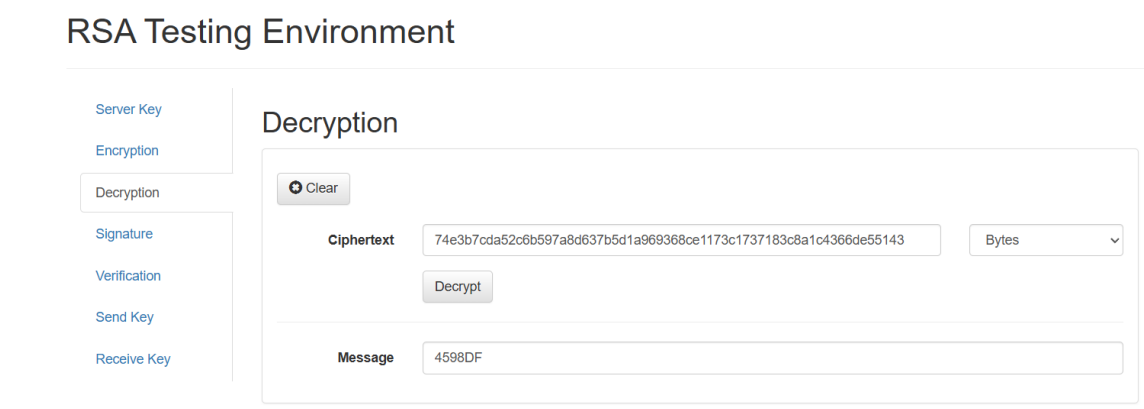
Перевірка



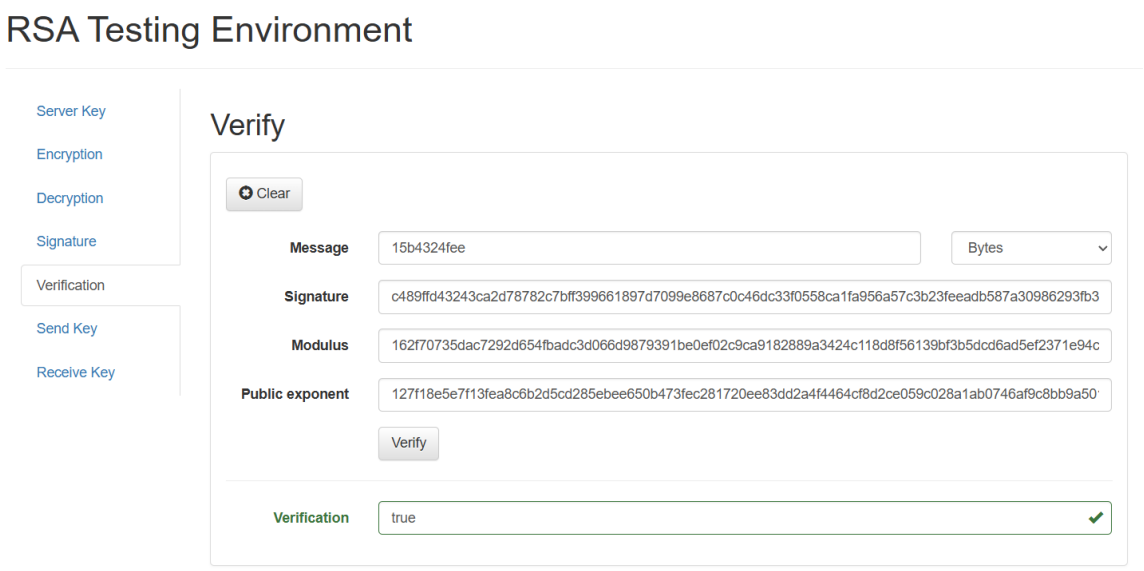

```
▶ Приклад 2: використання server key (256 біт)
   Server key size: 256 біт
   Server modulus (hex): 0x90829D6CAF6C2D76F691E6DC30FC86B8BAD615F148A6C6EF202573A70005FB5D
   Server public exponent (hex): 0x10001
   C = M^e mod server_m: 0x74e3b7cda52c6b597a8d637b5d1a969368ce1173c1737183c8a1c4366de55143
```



```
▶ Приклад 1: шифрування для великого модуля m
  M (десяткове): 4561119
  M (hex): 0x4598df
    m: 0x654efea76464a587b4eaa413301b5359f7c205f0a6b4f47b0c6fbc051a1f4df9a37ab47aa63c59c96ded00c12110de947d2f4f348ac56426749440f6165a77a3
    e: 0x4ac7dfdf0e05c7a7eb366084c8f450ebeb977693c5ed13b99b32c3bff89219bf8df226a89c5f6fdb15925d585bdba8378c3befac3fbb7e85637b1fbe04ac4efd
    C = M^e mod m: 0x33b28f9e0d9275f6796f7c0e17a7d8f88fc956873cda96cb75ee119a19f2b058420bad5f0e393611480c90a8f34407058617bc28a401677364c9b0d78158157f
```




```
▶ Приклад 4: підпис із заданими ключами
    M: 0x15b4324fee
    m публічного ключа: 0x162f70735dac7292d654fbadc3d066d9879391be0ef02c9ca9182889a3424c118d8f56139bf3b5dcd6ad5ef2371e94c3b30dd0c22676c62d30f99fb082cb0365
    e публічного ключа: 0x127f18e5e7f13fea8c6b2d5cd285ebee650b473fec281720ee83dd2a4f4464cf8d2ce059c028a1ab0746af9c8bb9a50128a32af4c6cbd194250cee89233106ad
    Підпис S = M^d mod (p*q): 0xc489ffd43243ca2d78782c7bff399661897d7099e8687c0c46dc33f0558ca1fa956a57c3b23feeadb587a30986293fb38b829c12c5b4b713fed5960ab29c84c
```



Висновок за результатами виконання роботи

Виконана робота являє собою повну практичну реалізацію криптосистеми RSA та демонстрацію її використання для засекреченого зв'язку (шифрування/розшифрування), електронного підпису та реалізації протоколу конфіденційного розсилання ключів з підтвердженням справжності.