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

КРИПТОГРАФІЯ

Комп’ютерний практикум №4

Варіант 7

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

Роботу виконав:
Студент 3 курсу
Групи ФБ-06
Кононець В. М.

Київ – 2022

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.

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 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, p1 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.

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

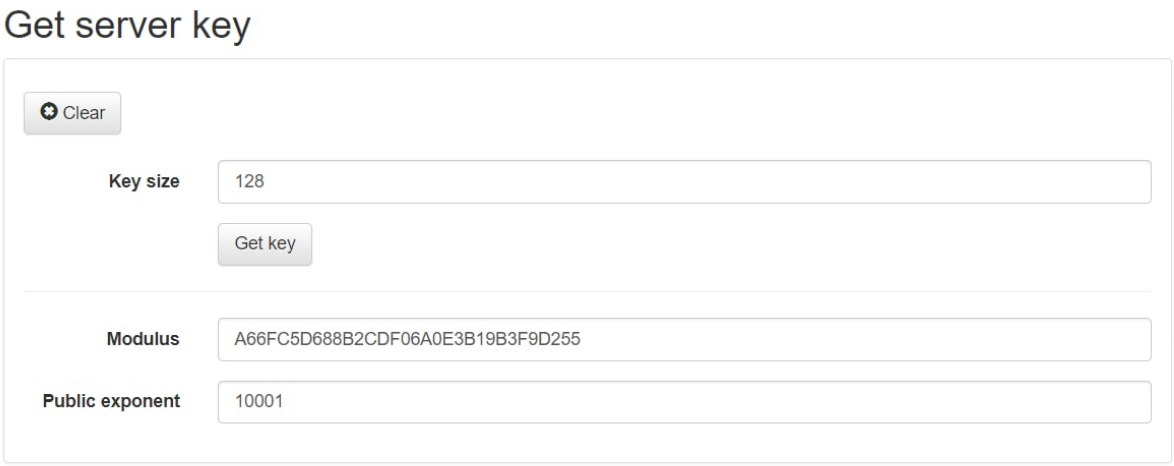
1. Із тестом Міллера-Рабіна проблем багато не виникло, трошки складно було зрозуміти програмно як знайти d та s із кроку 0 теста. Для перевірки чи правильно я зробив тест, використовував: https://planetcalc.ru/8995/
2. Згенерував числа, так, щоб pq <= p1q1:

p =
1148935766150127675326316174622661535319562673577836488614910828995683971950 91
q =
5890018389981813625480317121917945767902047908169793442386326460774698526 8011
p1 =
1129887886687507636096999265961409787367897557687835950003445539915452430 73851
q1 =
1133293021572194977348916595332210690419029240521908026236461086528143894 9334531.

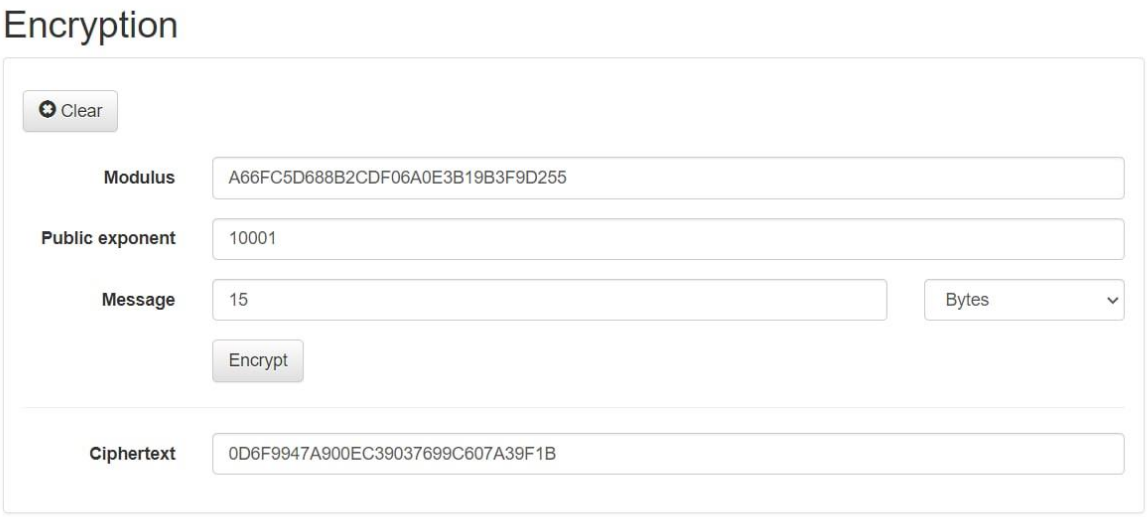
3. У загальному проблем на шляху до розв’язку не виникало, але були такі моменти, як дуже довга робота програми, і щоб її оптимізувати, потрібно було зробити алгоритм швидкого піднесення до степеня за модулем, що було для мене складним, так як ні алгоритму ні реалізації я не знав і у методичці не було пояснень до цього, тому наш товариш Гугл нам допоміг у цьому. Усі інші кроки подано на скріншоті:

```
Ключі для абонента A:
e : 29807854393030810786391920163968672727960849575831965965881404996111492056413203061221736397490294515251297595240712350426713788995972619132617405646066389
n : 67672527915320965323776865117463844233815710790346406712142066994886662891926451860280156574559722526000539596389691645218164199685538526192377512885340001
d : 36246591210100504034138644967636426525145859190013286213618113232982776239044134755914602312155502125477537243488785944469404166029644942386098849758406609
Ключі для абонента B:
e1 : 17995647370092455136307038792100834657164436671853838455004752143334058118861360292673149950109741360128986243729846522992299021640672445503843266863424 17
n1 : 12804940571419073856029106197101472604569568805394959488314569831560471942630314657184589953281589125270936586622993681916614289951515736366144152777448881
d1 : 60708134034973538255178762937407264470236041195274897938234812993543660859714455175173731076168714114328360254028035118822051158124914084392051863280972 53
Message: 25153325779863314304240823621654383680408100337432036202852110457294157030492293340995416400326954035883184739778113907861260305308901199243364181163459 14
Encrypted message: 28214579106701869816377622846104085041616328519352461399208843622451104247049220832658810171339568769377900631285141087174763966938068648605476997602515 38
Decrypted message: 25153325779863314304240823621654383680408100337432036202852110457294157030492293340995416400326954035883184739778113907861260305308901199243364181163459 14
Key k: 19745737808130181823024204283025903081508088408626833027355111512442746155650882587255053501020044640182133005598237658025597231413339400989724946124714469
(21230574080735292559942956801343621636733148141501065789562429781345729705691047298853841835794706164665659045855940208833255746302863707740787965392562 7, 28328495357159478789
Authentication succeeded.
(28214579106701869816377622846104085041616328519352461399208843622451104247049220832658810171339568769377900631285141087174763966938068648605476997602515 38, 25153325779863314304
Verification successful.
25153325779863314304240823621654383680408100337432036202852110457294157030492293340995416400326954035883184739778113907861260305308901199243364181163459 14
```

Тому я перейшов до перевірки на сайті.



Створив ключі для Абонента.

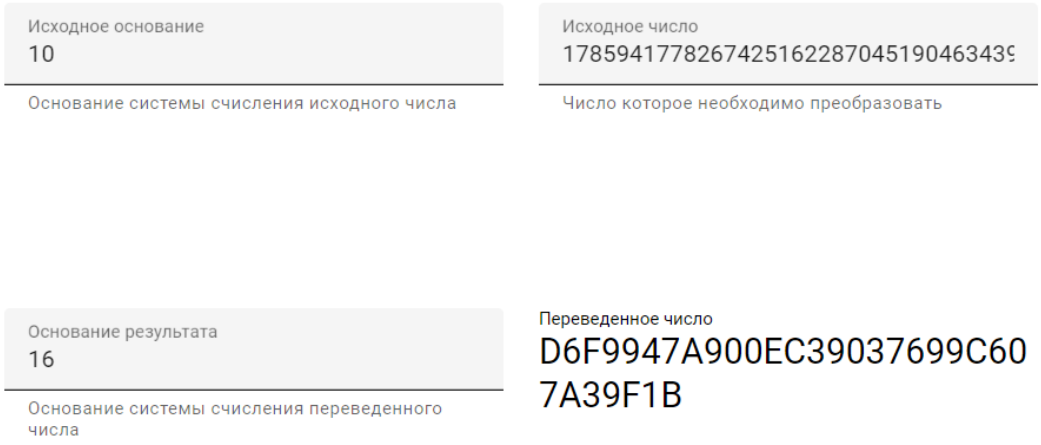


Зашифрував повідомлення 21, що у 16-ковому представленні буде 15

За допомогою свого коду отримав наступне:

```
n = 221232204883418220852024629632492491349
# A66FC5D688B2CDF06A0E3B19B3F9D255 (у 16-ковій)
e = int('10001', 16)
some_message = 21 #(15 у 16-ковій)
A = SubscriberKey('A', e, n, d=None)
print(A.encrypt(21))
# 178594177826742516228704519046343964 43
```

```
D:\Python\PycharmProjects\Python_3k\venv\Scripts
178594177826742516228704519046343964 43
```



Як бачимо, це те й саме число, що і на сайті.

Decryption

Clear

Ciphertext 0D6F9947A900EC39037699C607A39F1B Bytes

Decrypt

Message 15

Sign

Clear

Message 15 Bytes

Sign

Signature 79EA73E6BDEC30A59B0B0C5D8D1CDBE8

Verify

Clear

Message 15 Bytes

Signature 79EA73E6BDEC30A59B0B0C5D8D1CDBE8

Modulus A66FC5D688B2CDF06A0E3B19B3F9D255

Public exponent 10001

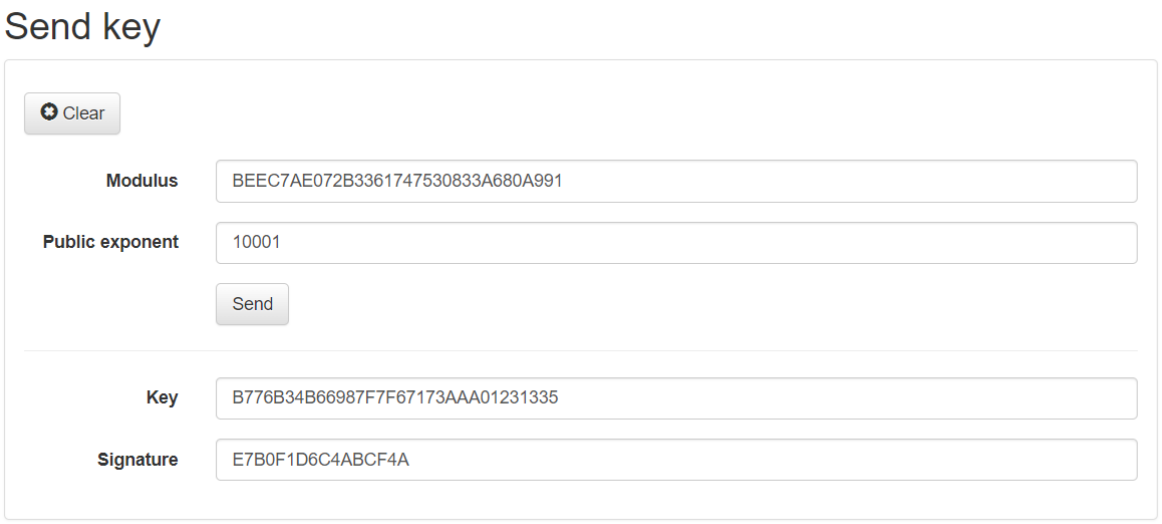
Verify

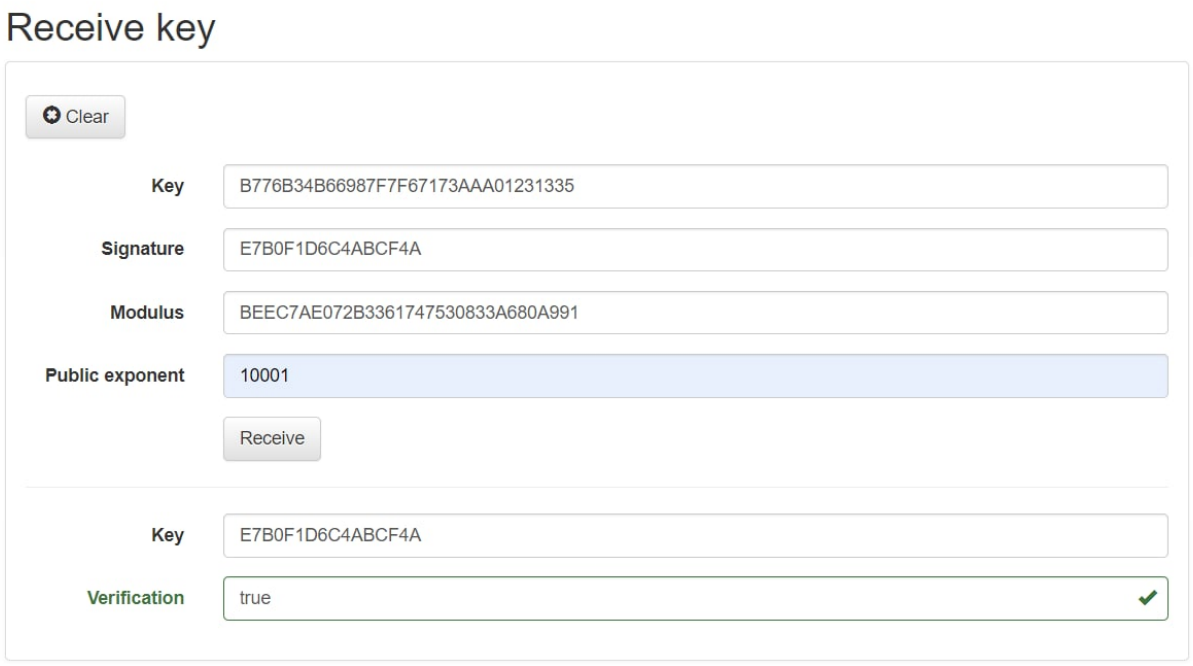
Verification true

```
signed_message = int('79EA73E6BDEC30A59B0B0C5D8D1CDBE8', 16)
print(A.verify(some_message, signed_message))
```

Вивід:

```
1
```





Висновки

У ході даної лабораторної роботи я ознайомився з тестами перевірки чисел на простоту, а найбільш із тестом Міллера-Рабіна, і методами генерації ключів для асиметричної криптосистеми типу RSA; практично ознайомився з системою захисту інформації на основі криптосхеми RSA, організував з використанням цієї системи засекречений зв'язок й електронний підпис, вивчив протокол розсилання ключів.