Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

# ЛАБОРАТОРНА РОБОТА №4

**з дисципліни «Криптографія»**

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

Виконали:

студент 3 курсу ФТІ

групи ФБ-81

Романченко Анатолій

Перевірили:

Чорний О.

Київ – 2020

**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 пошукувипадкового простого числа з заданого інтервалуабо заданої довжини, використовуючи датчик випадкових чисел татестиперевірки на простоту.В якості датчика випадкових чисел використовуйтевбудований генератор псевдовипадкових чисел вашої мови програмування. В якості тесту перевірки на простоту рекомендовано використовувати тест Міллера-Рабіна ізпопередніми пробними діленнями. Тестинеобхідно реалізовувати власноруч, використання готових реалізацій тестівне дозволяється.

2. За допомогою цієї функціїзгенерувати дві пари простих чисел q,p і q1, p1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб qp£q1p1; p і q–прості числа для побудови ключівабонента A, p1 і q1 –абонента B.

3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ q,p,d та відкритий ключ e,n. За допомогою цієї функції побудувати схеми RSA для абонентів Аі B–тобто, створити та зберегти для подальшого використання відкриті ключі ),(n1,e1)петасекретні d і d1.

4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів Аі 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 Мі знайти криптограму для абонентів Аи B, перевірити правильність розшифрування. Скласти для Аі Вповідомлення 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 nk<<0.

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().

Кожну операцію рекомендується перевіряти шляхом взаємодії із тестовим середовищем, розташованим за адресою

http://asymcryptwebservice.appspot.com/?section=rsa

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.

# Хід роботи

Для початку я генерував 256-бітне число та перевіряв його на простоту за допомогою теста Мілера Рабіна. Перевірялися лише непарні числа:

```
Miller Rabin failure(composite): 78232113505708313815429613712352134192418444899523486312415224208926905478385

Miller Rabin approve ( prime ) : 78232113505708313815429613712352134192418444899523486312415224208926905478387
```

З двох 256-бітних простих чисел можна згенерувати публічний та приватний ключі:

```
Alice public key:

       public exponent:
26268169379706994679390466011292013151084935028548116705052150583917603956744767297856146606445020170397811514959700675818108011978099794749792797580353 5

       modulus:
59383175673687696131217179398227486460592840185976966015107033762194582532878721233791831292334845500357890599704343540158346078908165199376497412536351 19

Alice private key:

       secret:
26268169379706994679390466011292013151084935028548116705052150583917603956744767297856146606445020170397811514959700675818108011978099794749792797580353 5

       modulus:
59383175673687696131217179398227486460592840185976966015107033762194582532878721233791831292334845500357890599704343540158346078908165199376497412536351 19
```

Далі на прикладі шифрування повідомлення «123456789»

Перевірив, що сайтик та моя програма однаково шифрує

```
Api and local encryptions are same? — True
```

Що моя програма правильно розшифровує

```
Local decryption test passed True
```

Також, перевірив сигнатуру повідомлення локально та за допомогою Api

```
Local verification: True

Api verification True
```

А також за допомогою Api та локально протокол конфіденційного розсилання ключів по відкритих каналах зв'язку з підтвердженням справжності відправника.

```
Bob got message 123456789 is valid message? - True
```

# Висновки

Під час виконання комп’ютерного практикума я ознайомився з тестом Мілера Рабіна перевірки чисел на простоту і методами генерації ключів для асиметричної криптосистеми типу RSA.

Також я дізнався про систему захисту інформації на основі криптосхеми RSA. Використував цю систему для засекреченого зв'язку й електронного підпису, реалізував протокол конфіденційного розсилання ключів по відкритих каналах зв'язку з підтвердженням справжності відправника.