

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

## **КРИПТОГРАФІЯ**

### **Комп'ютерний практикум №4**

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

Роботу виконали:  
Касаб О.Р.  
Косигін О.С.  
Групи ФБ-06

## 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.

## Порядок 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 і довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб  $p$  і  $q$  – прості числа для побудови ключів абонента  $A$ , і  $r$  – абонента  $B$ .  $q \neq p$ ,  $1 < q < p$ ,  $q \nmid p-1$

3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повертати та/або зберігати секретний ключ та відкритий ключ. За допомогою цієї функції побудувати схеми RSA для абонентів  $A$  і  $B$  – тобто, створити та зберегти для подальшого використання відкриті ключі  $e_A$ ,  $d_A$  та секретні  $d_A^{-1}$ ,  $e_B$ ,  $d_B$  та секретні  $d_B^{-1}$ .

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 . nk 0 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://asymcryptweb service.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:**

1. Під час виконання був обраний тест Міллера, який був рекомендований у методичці для реалізації

Також були використані функції вбудовані у пайтон функції для реалізації псевдовипадкових чисел

2. **значення вибраних чисел p, q, p1 , q1:**

p:

191772136514096483425832293639303549677550378395525470432119  
184961129579142549

q:

139904671983757115225113079900297343072858493355519498589637  
325770155831560029

p1:

226033581442392560027697793424882139296899596245901831433657  
557892460176128157

q1:

148394336250698434992550284706657293624827564811621342322804  
467548691337471861

**із зазначенням кандидатів, що не пройшли тест перевірки простоти у протокол ми їх не додали тому що їх багато і вони не несуть у собі смислового навантаження**

**і параметрів криптосистеми RSA для абонентів A і B**

e\_A: 65537

n\_A:

268298178546289591736121495991727807173121031820798051970306  
167373047491939157115246540294461811177935776550698078663348  
10378609801114799231529630841573921

d\_A:

201691355454782021088293601218005918427730196296981211016999  
159985472798348509852291652929638488242399022255642925525477  
70242217931591741088630423796370577

e\_B: 65537

n\_B:

335421032885120313200193272900575284014830383007337409743628  
553459095545140236532909289030793281573563603758012535808305  
06075122755809261027098843917290177

d\_B:


754040935272666978100683200275291157573660074896181097357657  
426814296453385270866058477130106828370258513976394103743792  
9446637695384040888222268065116913

3 та 4 пункти, приклади роботи кода та його обробка по вимогам звіту:

```
13 message: 178128042662910515065181430857464496826487405466273707949904368633834064642971
14 Зашифроване повідомлення: 26789921138845574933650180759559336768558829796729096896582881214407308066523672468618970401038860954459390709413518236228793901340495665732844851593863246
15 Розшифроване повідомлення: 178128042662910515065181430857464496826487405466273707949904368633834064642971
16
17 Ключ k: 230344327238633883356230784843048574452909207603940603109263666752964100637217545905624750661151838164796891247097610988967396900361795099319203719140445
18 Ключ k1: 2003509765107768030308858241403818397866909801970147823361523685618390762085266007101824302379774829702137349178838093545301917438438575927691214527805596
19 Ключ s1: 32080735200472960325213146647881823832253183151015505812464759417368268881343048427463211660180260489069073755996459908236800965688638978382700990043956558
20
21 Verify
22 Auth
```

Перевірка сайт у порівнянні з кодом:

## Get server key

 Clear

Key size

256

Get key


Modulus

975A520F21D133D2A05F167565FA4667D7047188DEC62B9924136D387F356B9F

Public exponent

10001

# Encryption

 Clear

Modulus

975A520F21D133D2A05F167565FA4667D7047188DEC62B9924136D387F356B9F

Public exponent

10001

Message

123321

Bytes

Encrypt

Ciphertext

3F758648BDF4DA9384016B4035ABBE8AFBCFF312B3034E15BEB43133A8415248

```
# 10:
# MOD = 975A520F21D133D2A05F167565FA4667D7047188DEC62B9924136D387F356B9F
# MSG = 123321
# e = 10001
# 16:
Serverkey_net = 68458822723064282581670949529857628029767980649954088402552635093188307413919
MSG_net = 1192737

print(encrypt(MSG_net, e, Serverkey_net))
28703357362456233546292521279654865941476018911048652171751107601302837875272
PS D:\BCĚ MOĚ\KPT\SEM5\Крипта\labs\crypto-22-23\cp4\lab_4_Kasah_fb-06_Kosygin_fb
```

From

Decimal

To

Hexadecimal

Enter decimal number

28703357362456233546292521279654865!

10

= Convert

✕ Reset

↕ Swap

Hex number

3F758648BDF4DA9384016B4035ABBE8AFB  
CFF312B3034E15BEB43133A8415248

16

## Sign

Message

123321

Bytes

Signature

43C1CF598E82191DA0A2C84D951577E41A0C0AE9A880B12B5EF2F3A2D0DD9887

## Verify

Message

123321

Bytes

Signature

43C1CF598E82191DA0A2C84D951577E41A0C0AE9A880B12B5EF2F3A2D0DD9887

Modulus

975A520F21D133D2A05F167565FA4667D7047188DEC62B9924136D387F356B9F

Public exponent

10001

Verification

true

✓

```
# Signature_16 = 43C1CF598E82191DA0A2C84D951577E41A0C0AE9A880B12B5EF2F3A2D0DD9887
Signature_10 = 30647393414517979894207132422897301279128145217284139372244085575598701648007
print(CheckSign(MSG_net, Signature_10, e, Serverkey_net))
28703357362456233546292521279654865941476018911048652171751107601302837875272
Verify
```

Робимо висновок що код працює так само як на сайті

**Висновки:** Під час виконання лабораторної роботи нами було оброблена інформація стосовно RSA криптосистем, а також пов'язаних з ним тестами на простоту текста(наприклад тест Міллера-Рабіна), організувати зв'язок за технікою RSA та обмін даними. Ще у живу зіткнулися з методами реалізації і роботи з електронним підписом, з шифруванням - розшифруванням інформації з потенційним захистом від втручання 3-й особи у лінію зв'язку