



Комп'ютерний практикум №4
Вивчення криптосистеми RSA та алгоритму
електронного підпису, ознайомлення з методами
генерації параметрів для асиметричних
криптосистем

Роботу виконали:

Біла Анастасія і Лета Яна,
студенти 3 курсу ФТІ НТУУ «КПІ»,
спеціальність «Кібербезпека», група ФБ-02

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 використовувати вбудований генератор псевдовипадкових чисел мови програмування Python. В якості тесту перевірки на простоту використати тест Міллера-Рабіна із попередніми пробними діленнями. Тести необхідно реалізовувати власноруч.
2. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p_1, q_1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $pq \leq p_1q_1$; p і q – прості числа для побудови ключів абонента А, p_1 і q_1 – абонента В.
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 А і В – тобто, створити та зберегти для подальшого використання відкриті ключі (e, n) , (e, n) та секретні d і d .
4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і В. 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 А и В, перевірити правильність розшифрування. Скласти для А і В 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.Реалізуємо функцію *test(p)* для перевірки на простоту числа, користуючись тестом Міллера-Рабіна із попередніми пробними діленнями із методичних вказівок.
- 2.Реалізуємо функцію *key_combination()*, що генерує дві пари простих чисел p , q і p_1 , q_1 довжини 256 біт при умові, щоб $pq \leq p_1q_1$.

Combination for A:

p: 69870608079018661601180799246745312521562105437529825401318468267276894639211

q: 67553771274572988924531841453353976879984408352672647655010173602171063373843

Combination for B:

p1: 82357176973860233037153705037657086267562250834061897690283334577624425056349

q1: 115426465953013127626989090887883401945315247109376207441527838013573418260803

- 3.Реалізуємо функцію *key_combination_rsa(p, q)*, за допомогою якої обчислюємо відкритий і секретний ключі: (n, e) та (d, p, q) .

A's keys:

d=399559082138800965245329774143628193960196240216967749086457180909850066287482
7076192011234309382909534184592523943986857723954683039338693779795964637027

p=69870608079018661601180799246745312521562105437529825401318468267276894639211

q=67553771274572988924531841453353976879984408352672647655010173602171063373843

n=472002307698535826876335701626939643795361899523927151999469383289200744062735
7956543087897149837455505580697517672903481508513592365586179285473899557873

e=596958022316664084558435054822137830537842589959146103708035209869712756652830
579169178496959762115932425343479340310478628077420218258622261109943796243

B's keys:

d1=64258682990243677152527323635838040175011940731783661192954382678815980124202
08297342250762472488453264443610654981703220526487588046107200760750030371277

p1=82357176973860233037153705037657086267562250834061897690283334577624425056349

q1=11542646595301312762698909088788340194531524710937620744152783801357341826080
3

n1=95061978839595549139428553007691607329543319404222634256309879704199059262391
51985132130051179601199611502486535653339410665801340716419609681458852988247

e1=72079830601265269847858778831729763005922549083658779072365859191417162456437
43401838218056422209911743631840087409825419729336980077106675287587067123877

4. Далі реалізуємо функції шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів А і В; організуємо роботу протоколу конфіденційного розсилання ключів з підтвердженням справжності по відкритому каналу за допомогою алгоритму RSA.

Отримаємо такі результати з перевіркою правильності зашифрування і розшифрування повідомлення та генерації ключа k:

```
Combination for A:
p: 69870608079018661601180799246745312521562105437529825401318468267276894639211
q: 67553771274572988924531841453353976879984408352672647655010173602171063373843

Combination for B:
p1: 82357176973860233037153705037657086267562250834061897690283334577624425056349
q1: 115426465953013127626989090887883401945315247109376207441527838013573418260803

A's keys:
d = 399590821388009652453297741436281939601962402169677490864571809098500662874827076192011234309382909534184592523943986857723954683039338693779795964637027
p = 69870608079018661601180799246745312521562105437529825401318468267276894639211
q = 67553771274572988924531841453353976879984408352672647655010173602171063373843
n = 4720023076985358268763357016269396437953618995239271519994693832892007440627357956543087897149837455505580697517672903481508513592365586179285473899557873
e = 596958022316664084558435054822137830537842589959146103708035209869712756652830579169178496959762115932425343479340310478628077420218258622261109943796243

B's keys:
d1 = 6425868299024367715252732363583804017501194073178366119295438267881598012420208297342250762472488453264443610654981703220526487588046107200760750030371277
p1 = 82357176973860233037153705037657086267562250834061897690283334577624425056349
q1 = 115426465953013127626989090887883401945315247109376207441527838013573418260803
n1 = 950619788395955491394285530076916073295433194042226342563098797041990592623915198513213005117960119961150248653565339410665801340716419609681458852988247
e1 = 7207983060126526984785877883172976300592254908365877907236585919141716245643743401838218056422209911743631840087409825419729336980077106675287587067123877

Our message: 2784704690640402619022374552253494066605330815550767380906811954006796310183032731046058315458317933860399114350207311954094331388303365013346976268
Encrypting: 19037849941694038812746559401448125789257393500223507513192692580058530495377302306610353901661544921562459198332686267916180458004498813239579708670
Decrypting: 27847046906404026190223745522534940666053308155507673809068119540067963101830327310460583154583179338603991143502073119540943313883033650133469762687
Phi: 4720023076985358268763357016269396437953618995239271519994693832892007440627220532163734305499311742864880598228271356967718311119309257537416025941544820

Checking message: Correct.
Checking key: Correct.

Process finished with exit code 0
```

5.Перевіримо роботу функцій за допомогою сайту, розташованого за таким посиланням: <http://asymcryptwebservice.appspot.com/?section=rsa> .

Get server key

✖ Clear

Key size

256

Get key

Modulus

8151DBE01E38BCFAD69AF5A7018AD9725E6E8FEF2556964E02D1173992A98077

Public exponent

10001

Message: A91

Encryption

✖ Clear

Modulus

8151DBE01E38BCFAD69AF5A7018AD9725E6E8FEF2556964E02D1173992A98077

Public exponent

10001

Message

A91

Bytes

Encrypt

Ciphertext

798CA917DA51ECA0E0405FE727738C3DF944A7B21223A316DECD71945CE27C1A

```
"""Checking...."""
n_hex = '8151DBE01E38BCFAD69AF5A7018AD9725E6E8FEF2556964E02D1173992A98077'
e_hex = '10001'
check_n = int(n_hex, 16)
check_e = int(e_hex, 16)
my_message_hex = 'A91'
my_message = 2705
print(to_encrypt(my_message, check_e, check_n))
# output:
# 54978380305847369047303250572451177412363843279305469386038678280741947866138
```

From

Decimal

To

Hexadecimal

Enter decimal number

549783803058473690473032505724511

10

= Convert

✖ Reset

↕ Swap

Hex number

798CA917DA51ECA0E0405FE727738C3D
F944A7B21223A316DECD71945CE27C1A

16

Значення співпадають, отже, функція працює вірно.

Decryption

✖ Clear

Ciphertext

798CA917DA51ECA0E0405FE727738C3DF944A7B21223A316DECD71945CE27C

Bytes

Decrypt

Message

0A91

Sign

✖ Clear

Message

A91

Bytes

Sign

Signature

5A6A2C0C427011ED2AD4C1F4B3E73C6B4CF2C4281C23151CAEAEBE282B9C61F2

Verify

✖ Clear

Message

A91

Bytes

Signature

5A6A2C0C427011ED2AD4C1F4B3E73C6B4CF2C4281C23151CAEAEBE282B9C61F2

Modulus

8151DBE01E38BCFAD69AF5A7018AD9725E6E8FEF2556964E02D1173992A98077

Public exponent

10001

Verify

Verification

true



```
signed_message = int('5A6A2C0C427011ED2AD4C1F4B3E73C6B4CF2C4281C23151CAEAEBE282B9C61F2', 16)
print(to_check(my_message, signed_message, check_e, check_n))
```

True

Process finished with exit code 0

Функція повернула вірне значення **k**, тобто все працює відмінно.

Висновки:

У ході виконання комп'ютерного практикуму ми практично ознайомилися із асиметричною криптосистемою типу RSA, створивши тест перевірки чисел на простоту алгоритму Міллера-Рабіна, реалізували засекречений зв'язок й створення електронного підпису, вивчили протокол розсилання ключів, перевірили свою роботу за допомогою вказаного вище сайту.