

**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**

Виконав:
Іванілов Ігор
Березовський Максим
Варіант 12
Група:
ФБ-06

Київ - 2022

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.

Хід 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 $1 < p, q$ довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $p \neq q$; p і q – прості числа для побудови ключів абонента А, $1 < p < q$ – абонента В.

Ось які числа вдалось створити так перевірити за допомогою тесту Міллера-Рабіна:

```
p1 76359642116807025490125207355526350059270391187441455021752396388666507470793
q1 58309526249452777231739214274484874053966235044824936805329617731296510417127
p2 109308004744107177220974944847356616379197318615147233568326696398477506954011
q2 89914520439454877512413578544902776404014761058769747551219190452650951080269
```

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) , (d, n) і та секретні d і d_1 .

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 і 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 $0 < k < n$.

Скріни працездатності коду:

```
Для абонента А
e : 1407620411423288915857229295366903674291910926445971954919040386343181783924574991738488263148999033410591681107806021488223951364040884141029699831715445
n : 4452494556408779084258312869603697031008295117476224294496545627377027348310338485769321126009084082657305365323234401167193068456297691976142094199471711
d : 1099595129575089802984485240724557953805966933758892250431590819722782379279233255949049184767490101727111262953207880316023328402728400261024293064755133
Для абонента В
e : 204811893583550684429692437607548583905732662001915464778388298983101700977230408848546833433953838814113217930018363368513690085459238524261088778581930391
n : 982837682676005550432683012968450002063636579945582968438224261794513004141151912406939985281373279420507077332965517035111616381444739141583420152508959
d : 6568346100952624056266834396742210811238251271126988172430261783311266766968045660204260222652389800041945691349374625464123853520056450642910686882064231
Повідомлення : 2591887413868191070467170238435245923355078316406777793896740560211539635510704841155610771300538905694217943167059444800065747454789259181608388395876983
Зашифроване повідомлення : 2282724904628511195975786157929541580312845240883279518198621433295121159639529587961817839462279631742454699155802138745856019584014543912246839486373645
Розшифроване повідомлення : 2591887413868191070467170238435245923355078316406777793896740560211539635510704841155610771300538905694217943167059444800065747454789259181608388395876983
Ключ k : 392330603223102831003635577459937878618575477664257520643298579893031553590332329204015418379344706752115393850102682457656142265345743811609830314956702
Авторизований користувач
Ключ k1 : 70121923851136907378578637501300630101100784488417689545023817944568117102781822714025882918070413687806006017614828025224833909094888764592168621896336514
Ключ s1 : 8717624337749554910000279326513131501862536786674443564969459975002677583038840380659697169449439218749391286972890671892457785766873354737564132690894886
Верифікація пройшла успішно
```

Роботи з сайтом:

Get server key

✖ Clear

Key size

255

Get key

Modulus

445A72E51B24B27FB894EAD817005587337E54113E55CBA55B07BF62B88C7E63

Public exponent

10001

Encryption

✖ Clear

Modulus

445A72E51B24B27FB894EAD817005587337E54113E55CBA55B07BF62B88C7E63

Public exponent

10001

Message

11

Bytes

Encrypt

Ciphertext

06C1B09C7B95F434366A58BAF1FF851B28BFC4A4B2CC8EF4DDE058FEF054482C

Decryption

Ciphertext

06C1B09C7B95F434366A58BAF1FF851B28BFC4A4B2CC8EF4DDE058FEF05448

Bytes

Message

11

Sign

Message

11

Bytes

Signature

3927E503BC423BC0D119590F2B664075329DCBE3F9851D548141F81EF6135849

Verify

Message

11

Bytes

Signature

3927E503BC423BC0D119590F2B664075329DCBE3F9851D548141F81EF6135849

Modulus

445A72E51B24B27FB894EAD817005587337E54113E55CBA55B07BF62B88C7E63

Public exponent

10001

Verification

true

```
n_server = 30917082915262018153573483920186964487956177587076973408317738523290518716003
e_server = 65537
message_server = 17
encrypt_server = encrypt(message_server, e_server, n_server)
print(encrypt_server)

sign_server = 25852320005391753845252256083997989028406097176929540733642651573387602909257

verify_server = check_sign(message_server, sign_server, e_server, n_server)
print(verify_server)
```

```
3056097501125641061195018314715562293168150133441813213678181391295898208300
Верифікація пройшла успішно
```

Висновки

Впродовж цієї роботи ми навчилися працювати з великими простими числами, а саме перевіряти її простоту та використовувати її при подальших діях. Також познайомилися на практиці з RSA та створили свій приклад роботи захищеного трафіку. Ці знання допоможуть нам у подальшому розвитку як спеціалістів, а також роботі чи навчанні.