

КРИПТОГРАФІЯ

КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

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

Виконав

ФБ-12 Сущенко Олександр

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 p, q і p_1, q_1 довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $pq \leq p_1q_1$ $p < q$; $p_1 < q_1$ – 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_1, n_1) 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 $0 < k < n$.

Хід роботи:

Реалізовуємо необхідні функції та отримуємо результат генерації ключів для абонентів:

```
Відкритий ключА (e, n): (139856974317497316754419755207265602247387147338813363836732473933785876915340100648910768977191019659316354999924526931837078917453439520413743262096911,
Секретний ключА (d, p, q): (95514439488287950287874492660980183297122281403088580407289301194112621785065787508146018339703002196097141230012469891433284473834559669243784345410991,
Відкритий ключВ (e, n): (12516272387556696211285661832907898552026012617905901116757637064081602901897026899198262768167014491900821798496659339696334788982654574994658230264487,
Секретний ключВ (d, p, q): (21297425474184475417311251234146326418074472451320405004778566598232856280040071018960725482081288835683725803817931190578306348370438048081672654075
```

Відкритий ключА (e, n):

(139856974317497316754419755207265602247387147338813363836732473933785876915340100648910768977191019659316354999924526931837078917453439520413743262096911, 299964712738848086959836632989880467227322602768516735408243401216270204944028036723324806042055652390150974615338208577906891795652745435531466278997653)

Секретний ключА (d, p, q):

(95514439488287950287874492660980183297122281403088580407289301194112621785065787508146010339703002196097141230012469891433284473834559669243784345410991, 31392130125537631667930448304987668245045383763762062358327012456296937055473, 9555411230116732809077817784517740600903110157197759180361106021211732210661)

Відкритий ключВ (e, n):

(1251627238755669621128566183290789855202601261790590111675763706408160290189702689919826276816701449190082179849665933969633478898265457499465823026448785, 6283065955613953792595455526834280858789629311847794822745805987533580202794518446502464735551391888405017793368182628421846350561823644460148269369790639)

Секретний ключВ (d, p, q):

(2129742547418447541731125123414632641807447245132040500477856659823285628004007101896072548208128883568372580381793119057830634837043804808167265407508841, 66703013237530860283504751456282255986315078390058947343412101309752343474147, 94194634554811207894024303222266909457649795877167430498404359734153813597637)

Також отримуємо певну кількість кандидатів, що не пройшли та виводимо 10 перших:

```
Кандидати, що не пройшли (247): [31392130125537631667930448304987668245045383763762062358327012456296937055139, 313921301255376316679304483049876682450453837637620623583270124562
```

Далі вибираємо деякий ВТ, шифруємо та розшифровуємо його:

```
ВТ: 4561119
ШТ: 5126870989888806682030634870659928821248944031054518927051468552918767187410501815571159380093128404357057502477996306977475794399608213935279729611490
Розшифрований: 4561119
```

Потім для цього ж ВТ генеруємо підпис та перевіряємо його:

```
Підпис S: 18317468890342328838227230150585201798943095893295188948685149486645842315514182439236233467471000382476012645384139702383001474789707959156126472438668
Перевірка підпису: True
```

Для розсилання ключів спочатку генеруємо k , $0 < k < n$. Формуємо повідомлення $(k_1 S_1)$, де

$$k_1 = k^{e_1} \bmod n_1, S_1 = S^{e_1} \bmod n_1, S = k^d \bmod n$$

Далі абонент В отримує повідомлення та виконує наступні дії:

$$k = k_1^{d_1} \bmod n_1, S = S_1^{d_1} \bmod n_1, \text{перевіряє підпис } k = S^e \bmod n$$

Результат:

```
k: 1174014691070418236828783371861174585518165316667588386948061081491727156993809345616732167457810935596335537403913738239706062209009047340600967781078
k1: 5384173273326839021208022704936539915911440544991006612307036442838220778569168139787305048299681941607211406758522053761734230995462276099850433895590
S: 170340238042577972876858795764358624143352473204923150413856177807213102191821264305687404992944263084066263137979914763549542515209261509538264772132
S1: 72384387040535659042491749877821004145122013924082561619676593473514593705505973650441311155714864968435823874666464037581455543241790494328385518359
Відправлений ключ (k1, S1): (53841732733268390212080227049365399159114405449910066123070364428382207785691681397873050482996819416072114067585220537617342309954622760998504
k: 1174014691070418236828783371861174585518165316667588386948061081491727156993809345616732167457810935596335537403913738239706062209009047340600967781078
S: 170340238042577972876858795764358624143352473204923150413856177807213102191821264305687404992944263084066263137979914763549542515209261509538264772132
check: True
Отриманий ключ (k, check): (117401469107041823682878337186117458551816531666758838694806108149172715699380934561673216745781093559633553740391373823970606220900904734060096
```

Перевірка:

Encryption

✖ Clear

Modulus

654fe476464a587b4eaa413301b5359f7c205f0a6b4f47b0c6fbc051a1f4df9a37ab47aa63c59c96ded00c12110de947d2f4f348ac56426749440f6165a77a3e08c3bfbf7e85637b1fbe04ac4efd

Public exponent

4ac7dfdf0e05c7a7eb366084c8f450ebef977693c5ed13b99b32c3bff89219bf8df226a89c5f6fdb15925d585bdba8378c3befac3fbb7e85637b1fbe04ac4efd

Message

4598df

Bytes

Encrypt

Ciphertext

33B28F9E0D9275F6796F7C0E17A7D8F88FC956873CDA96CB75EE119A19F2B058420BAD5F0E393611480C90A8F34407058617bc28a401677364c9b0d78158157f

```
BT: 0x4598df
m: 0x654fe476464a587b4eaa413301b5359f7c205f0a6b4f47b0c6fbc051a1f4df9a37ab47aa63c59c96ded00c12110de947d2f4f348ac56426749440f6165a77a3e08c3bfbf7e85637b1fbe04ac4efd
e: 0x4ac7dfdf0e05c7a7eb366084c8f450ebef977693c5ed13b99b32c3bff89219bf8df226a89c5f6fdb15925d585bdba8378c3befac3fbb7e85637b1fbe04ac4efd
ШТ: 0x33b28f9e0d9275f6796f7c0e17a7d8f88fc956873cda96cb75ee119a19f2b058420bad5f0e393611480c90a8f34407058617bc28a401677364c9b0d78158157f
```

Get server key

✖ Clear

Key size

256

Get key

Modulus

CD2393443A84B3898BAAFAC7A4F56DCF363542A01FBE07DE229C91998BBE8B85

Public exponent

10001

Decryption

✖ Clear

Ciphertext

c30a1bb10e068fb0d73261c5083ccdc34b9e7fcd8cfd0b5e6faa3b8449a98c5

Bytes ▾

Decrypt

Message

4598DF

```
m: 0xcd2393443a84b3898baafac7a4f56dcf363542a01fbe07de229c91998bbe8b85
e: 0x10001
WT: 0xc30a1bb10e068fb0d73261c5083ccdc34b9e7fcd8cfd0b5e6faa3b8449a98c5
```

Sign

✖ Clear

Message

738c4410

Bytes ▾


Sign

Signature

067D48439EF451F4023AE1DD1EB16936D9C273FD9CA466E82E6B5F1C314A4297

```
M: 0x738c4410
S: 0x67d48439ef451f4023ae1dd1eb16936d9c273fd9ca466e82e6b5f1c314a4297
Verify: True
```

Verify

 Clear

Message

15b4324fee

Bytes

▼

Signature

c489ffd43243ca2d78782c7bff399661897d7099e8687c0c46dc33f0558ca1fa956a57c3b23feeadb587a30986293fb3

Modulus

162f70735dac7292d654fbadc3d066d9879391be0ef02c9ca9182889a3424c118d8f56139bf3b5dcd6ad5ef2371e94c

Public exponent

127f18e5e7f13fea8c6b2d5cd285eb6e650b473fec281720ee83dd2a4f4464cf8d2ce059c028a1ab0746af9c8bb9a50

Verify

Verification

true

✓

```
M: 0x15b4324fee
m: 0x162f70735dac7292d654fbadc3d066d9879391be0ef02c9ca9182889a3424c118d8f56139bf3b5dcd6ad5ef2371e94c3b30dd0c22676c62d30f99fb082cb0365
e: 0x127f18e5e7f13fea8c6b2d5cd285eb6e650b473fec281720ee83dd2a4f4464cf8d2ce059c028a1ab0746af9c8bb9a50128a32af4c6cbd194250cee89233106ad
S: 0xc489ffd43243ca2d78782c7bff399661897d7099e8687c0c46dc33f0558ca1fa956a57c3b23feeadb587a30986293fb38b829c12c5b4b713fed5960ab29c84c
```

Висновок

Під час виконання цієї лабораторної роботи я набув навичок пошуку випадкового простого числа, написання тестів для його перевірки, генерації ключових пар для криптосистеми RSA, її використання для шифрування та дешифрування повідомлення, а також роботи з цифровим підписом і протоколу розсилання ключів.