

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КІЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

КРИПТОГРАФІЯ
КОМП'ЮТЕРНИЙ ПРАКТИКУМ №4

Вивчення крипtosистеми RSA та алгоритму електронного підпису;
ознайомлення з методами генерації параметрів для асиметричних
крипtosистем

Виконали:
студенти групи ФБ-32
Кошикова Дар'я
Сажко Олена

Мета роботи:

Ознайомлення з тестами перевірки чисел на простоту і методами генерації ключів для асиметричної крипtosистеми типу RSA; практичне ознайомлення з системою захисту інформації на основі крипtosхеми 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 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 $pq \neq p_1q_1$; $p \neq q$ – прості числа для побудови ключів абонента А, $p_1 \neq q_1$ – абонента В.
3. Написати функцію генерації ключових пар для RSA. Після генерування функція повинна повернати 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) , (n_1, e_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 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

Згенерували p , q , p_1 , q_1 :

```
p = 71876647656307248714409587053377957278616954632842566787505358976909620194139
q = 70798927610520525231292353932221728480580312529980744989516914667068324521493
p1 = 65047982127594580230212490056993439771450785755760407399238693462963124491449
q1 = 112728816273313520873810346569451580772568084268516937208776119348111697668021
```

Ці числа пройшли тести простоти (Міллер–Рабін + ділення малими простими).

Вони використовуються для побудови модулів RSA, а успішний прохід тестів означає, що вони ймовірно є простими.

На основі p і q для абонента А згенеровано відкритий та закритий ключ RSA:

```
abonent A:
public key (e, n): [65537,
50887895743057866702955682775033788153038476182639808120679509675177294755768
05183750956082690456273964183950885775001053248926294491378372385916938129527
]
private key (d, p, q):
[4615760505433686140893236740409056772139970118676508527600887212781961772367
58934340149606723701071080334812376344567886940031420125203927994437590160503
3,
71876647656307248714409587053377957278616954632842566787505358976909620194139
,
70798927610520525231292353932221728480580312529980744989516914667068324521493
]
```

Пара ключів дає можливість шифрувати дані (через e , n) та створювати цифрові підписи (через d)

Аналогічно, для абонента В були сформовані власні ключі RSA на основі p_1 і q_1 .

```
abonent B:
public key (e, n): [65537,
73327820262113909783984250947353415379560611609624825023297782865295154378939
63514111260650329555529464765252646455718446840743853302067393662764855252429
]
private key (d, p, q):
[5068846973975840296235467782577706908983069697652979916734759688643768220744
04039332719662026298943504851568429858976953367588358092934775882820564824771
3,
65047982127594580230212490056993439771450785755760407399238693462963124491449
,
11272881627331352087381034656945158077256808426851693720877611934811169766802
1]
```

Далі перед шифруванням і підписанням абонент А виконує базові операції RSA над повідомленням:

```
abonent A
```

```
encrypted:  
4719632524780056642307664516803053009545526780021065991232337797358999736354  
49168770973480764557045198723010787912736443560795871344750251338359253410238  
decrypted: 12345678901234567890  
signature:  
43938553941423879005974537567031186562366845441370444956991917376768304416246  
37817189249162012787550512562395184800154041385785831334635174773875019833464  
check: True
```

Шифрування показує, що публічний ключ працює правильно, а розшифрування — що приватний ключ коректно відновлює повідомлення. Підпис створений приватним ключем, і значення check: True означає, що публічний ключ підтвердив його справжність.

Після надсилання зашифрованого ключа і підпису абонент В виконав розшифрування й перевірку:

```
abonent B  
encrypted:  
18206405215274783332775879432580972915011659643441824159561380482483931875598  
95854312180703446659400730462407340645511582344032438413027037463440308584289  
decrypted: 12345678901234567890  
signature:  
50644879818526040929591299342065471982213365440121423813023502200215668733087  
69995735218738774127211905540551590093257882106318620290803584481558421283033  
check: True
```

Публічний ключ абонента В успішно шифрує повідомлення, а приватний правильно його відновлює. Підпис сформований приватним ключем абонента В, і check: True підтверджує, що він є дійсним.

Після надсилання зашифрованого ключа та підпису абонент В розшифрує дані й перевіряє підпис абонента А. Результат роботи програми:

```
exchange protocol (A -> B)  
message k = 12345678901234567890  
signature confirmed
```

Абонент В успішно розшифрував значення k, а підтверджений підпис показує, що дані автентичні й не змінені.

Перевірка з сайтом:

1. Шифрування повідомлення зі згенерованим у себе ключем

Encryption

Modulus	978790B33A5157F8F092561AF780F08B814256EDB90295162159F8806F37050FEC4CEF4467B2E4B3CECF691
Public exponent	10001
Message	crypto lab4
	<input type="button" value="Encrypt"/>
Ciphertext	60EF623CA141B31BBEFD6E1E6523576ED05B01C6C7F2495EF5BA55BFE4F74AAAD9BA55541375576F65264

Перевірка шифрування своїм ключем
modulus (n): 978790B33A5157F8F092561AF780F08B814256EDB90295162159F8806F37050FEC4CEF4467B2E4B3CECF691D9E2FC3309697FB3790DBBED557E84589098F7059F
експонент(e): 10001
Наше шифрування: 60EF623CA141B31BBEFD6E1E6523576ED05B01C6C7F2495EF5BA55BFE4F74AAAD9BA55541375576F65264FF3EE0D09A49983D6A2AD5267DE0AE21707FAAD485E

2. Розшифрування повідомлення зашифрованого ключем з сайту

Decryption

Ciphertext	9272DA8FEB6636CF8E10AD88C3D70480C2E6EEED35A9003F800A069B98A471	<input type="button" value="Text"/>
	<input type="button" value="Decrypt"/>	
Message	crypto lab4	

Шифрування з відкритим ключем сервера
Encrypted (server) = 9272DA8FEB6636CF8E10AD88C3D70480C2E6EEED35A9003F800A069B98A47153

3. Перевірка підпису з сервера

Sign

Message	crypto lab4	<input type="button" value="Text"/>
	<input type="button" value="Sign"/>	
Signature	59577B516FCE89EDDD868E6B6843094349FA74B8F86EE1AAA538AB3AE0A454AC	

введіть підпис, виданий сайтом: 59577B516FCE89EDDD868E6B6843094349FA74B8F86EE1AAA538AB3AE0A454AC
Перевірка підпису, зробленого сервером
Перевірка підпису сервера: True

4. Перевірка нашого підпису

Verify

Message	crypto lab4	Text
Signature	E05C19A968CBA43A20674ADE2DF9D0F87AF5BD3B4638CCEEF645DDA6E1B7177467F0BA1A07BFD378B8D	
Modulus	978790B33A5157F8F092561AF780F08B814256EDB90295162159F8806F37050FEC4CEF4467B2E4B3CECF69I	
Public exponent	10001	
<input type="button" value="Verify"/>		
Verification	true	<input checked="" type="checkbox"/>

Наш підпис
My sign = E05C19A968CBA43A20674ADE2DF9D0F87AF5BD3B4638CCEEF645DDA6E1B7177467F0BA1A07BFD378B8D88A0E9D44661015F21A2657B7948AE013CA3302D5842

Висновки:

Під час цієї лабораторної роботи ми ознайомилися з роботою асиметричної криптографії на прикладі RSA. Ми навчилися перевіряти числа на простоту, генерувати великі прості числа для ключів, шифрувати та розшифровувати повідомлення, а також створювати й перевіряти цифрові підписи. Особливо цікаво було побачити, як працює протокол безпечної розсилки ключів і як можна передавати секретну інформацію через відкритий канал.