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

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 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* ≤ *p*1*q*1 ; *p* і *q* –  
прості числа для побудови ключів абонента *А*, *p*1 і *q*1 – абонента *B*.

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 *B* – 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 *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 *А* и *B*, перевірити правильність розшифрування.  
Скласти для *А* 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 0 < *k* < *n*.

**Хід роботи**

**Порядок дій програми на локальній машині.**

1. **Генерація ключів для A та B**

При умові що n1(B)>= n(A)

A. Дані ключів

n = 7550BBB555BD08071F2A565FC220266111DB76D4D87311CC519BE64A5F1B99074F9B02EEE7BD10D28D71E34B7F974763587BB3872307D7DA059BBA1B82FEF445

e = 62EB9721930647145E0B31BFB89B8031DF15ED4BA3B6F8D2D4FD8FE1BAF5443E30B4777B3BCED150F66F4782C213A8C3CD0BF650CE404EB0C801BB860DE22BC3

d = 4EA43B645C9581B48B3724C190CE61182BE84423E6795D5D61C064CD6105D5D5F1C3265B8DB82D5DA1675BAEE155D5C79F51EC406D003393F788A63C768C993B

p = C5D65F8891768E2DF8118B58A2B69EE845C0669A5D046627025F023998E09AE3

q = 97CE1D9306A15B9F5C381498F4C8DDAFE4E6AF35809EC49C7F4CB5ABF79794B7

B. Дані ключів

n = E4C7E49E662166D654632C1B9C08FEBBA94F92B95707AF91B6D7DF8B660412CA8AE2054DEAD6430526C9EE0825D1D4AEB674377277886B35DBEFC649AF771A99

e = 5B589DF29FD9FA45BB82CFDC9BC787B89B55A4A18B0A440E2A26EC8F2FC8A7CB1C185265E48B61F446B0A8183AC4C8FE63094686A0128CF3C098652BB3F2D1E7

d = CC5959D4F47BA3493EA288C2A3C7019B450EB2CAA2D2BD0B2862A6D53DF7FE0FA674BA411BEC126270D43B2CBD6C588464EB4B4DC7D2A4562E4D80C28C97B9BF

p = F51519B220247EC2FF4694578C02400635A9EC4CB8E8B289CAD38DE40744FAC3

q = EEF8E403404C5F4B98B60ABE97C2F2F51A150F5C36613465B0ADC89085B26773

1. **Генерація відкритого повідомлення**

Генеруємо число k в діапазоні від 1 до n – 1.

3389461AC191DB7406273F0D4E0B06F46D22EE9079564A1D1E35788678E4B6A65EED47AEDC5DE1E931A7AE63F9AC940B98ECD98F9C904D338D8E29099AF80C72

1. **Відсилка повідомлення від A до B**

k1 = 6AB959381BCC04FFF28EB53803DB96BBA0FC2F8493FB9BC727BF912947D920D5196629AC5F488386F13A7A61B545DC9D6B8A8AA0B53F1777E689F5D50DCDE09C

S1 = A9B64A944A26F1027EEAB823AA8FA0536A2E0B067265D1B92E734054FD00BD50DA52BE01C02EC2F4EE6C641B240526F256D582938991323C238981297D9C20

1. **Отримання повідомленння для B від A**

Співпадіння ключів (Verification) true

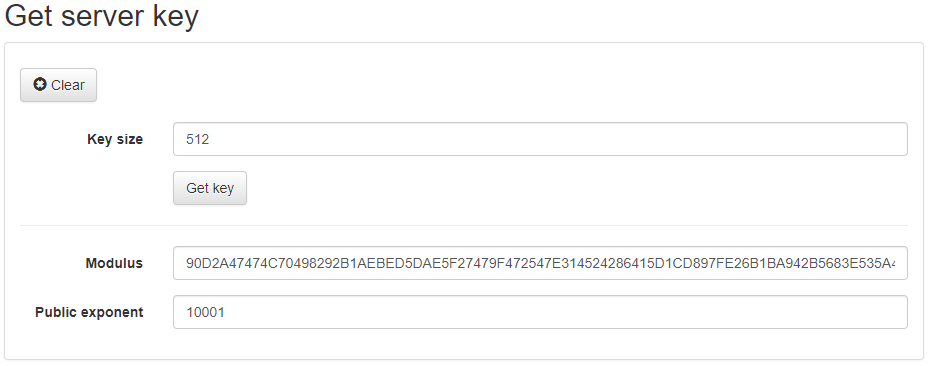
Розшифроване повідомлення

3389461AC191DB7406273F0D4E0B06F46D22EE9079564A1D1E35788678E4B6A65EED47AEDC5DE1E931A7AE63F9AC940B98ECD98F9C904D338D8E29099AF80C72

Список простих чисел кандидатів можна знайти у файлі numbers1.txt.

**Порядок дій програми при роботі з сайтом для тестування RSA.**

1. **Отримуємо на сайті відкритий ключ для B**



n=

90D2A47474C70498292B1AEBED5DAE5F27479F472547E314524286415D1CD897FE26B1BA942B5683E535A42364BBE945C635CE3EE8A2CCC389BD55BB8BB90953

e= 10001

1. **Генеруємо відповідний ключ для A**

При умові що n1(B)>= n(A)

n = 8463B05FCA744C62D9A46CDF9B1F1C01D7852545FABE3DE984B2596F0DDD63AAC3CE46D99FBF907C58AB650D07B62A71BC3505E2C9B1EAC79458A92EE5F55799

e = 8264B2B8FCB85C40B88A9B4F2A0317D09AA728DB77ECF900D93189A554887C38EBFF4C91837778053C87359EF25F76DB64F79BDAD3829DFA9783E893E5E2BAB7

d = 9414832DC848BAC08A7AF237CD5E6249180FA6B8DBC1E1B546DB1CB1C0E8308EC3BB0ADF75D6303534BFF1A417F9EB5C62A8F701FD4ADC1FBACC4688AC20477

p = 99A0B3BADEF9D3F3372D07068A767C8B09F5A84D6BA43C12369EFC3EBDB56F57

q = DC9BF2A768E8510DBBBF8700F41E2AC711D31631C27B0E4596E0AB4224DB4A8F

1. **Генеруємо відкрите повідомлення**

Генеруємо число k в діапазоні від 1 до n – 1.

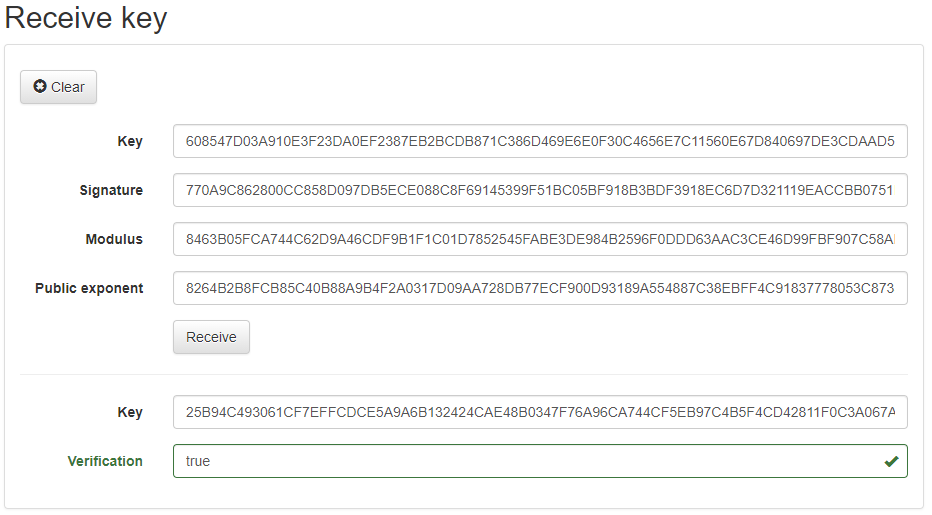
25B94C493061CF7EFFCDCE5A9A6B132424CAE48B0347F76A96CA744CF5EB97C4B5F4CD42811F0C3A067A10F2267BCFF87E327F143685B244FD8EE5E4B60E9863

1. **Створюємо повідомлення для відправки**

k1 = 608547D03A910E3F23DA0EF2387EB2BCDB871C386D469E6E0F30C4656E7C11560E67D840697DE3CDAAD58C0999FF9AE1A6EF50B56F4CAD572EB6F9C5817F0

S1 = 770A9C862800CC858D097DB5ECE088C8F69145399F51BC05BF918B3BDF3918EC6D7D321119EACCBB0751FBDD86FC755BB4AD8EC01CE78F0FC38F5A1A10D8536D

1. **Отримуємо відкрите повідомлення через сайт**



**Висновок**

В ході виконання лабораторної роботи ми ознайомились з тестом перевірки числа на  
простоту, методами генерації ключів для асиметричної криптосистеми типу RSA, системою  
захисту інформації на основі криптосхеми RSА. Було реалізовано відповідні функції для роботи з RSA такі як тести чисел на простоту та генерацію випадкових чисел, схема Горнера швидкого піднесення до степеня та сім високорівневих процедур: GenerateKeyPair(), Encrypt(), Decrypt(), Sign(), Verify(), SendKey(), ReceiveKey(). А також практично було реалізовано протокол передачі ключів RSA.