Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

Криптографія

Комп’ютерний практикум №4

Варіант 1

Виконав
студент гр. ФБ-03 Антіпов Данило
Перевірив
Чорний Олег Миколайович

Київ — 2022

Мета та основні завдання роботи

Ознайомитися 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 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб pq ≤ p1q1 ; p і q – прості числа для побудови ключів абонента A, 1 p і q1 – 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 A і B – тобто, створити та зберегти для подальшого використання відкриті ключі (e,n) , (,) 1 n1 e та секретні d і d1 .

4. Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів A 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 A и B, перевірити правильність розшифрування. Скласти для A 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 $0 < k < n$.

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().

Кожну операцію рекомендується перевіряти шляхом взаємодії із тестовим середовищем, розташованим за адресою

http://asymcryptwebservice.appspot.com/?section=rsa.

Наприклад, для перевірки коректності операції шифрування необхідно а) зашифрувати власною реалізацією повідомлення для серверу та розшифрувати його на сервері, б) зашифрувати на сервері повідомлення для вашої реалізації та розшифрувати його локально.

Хід роботи

1. Спочатку мною були реалізовані декаілька функцій: пошук НСД, простий тест числа на простоту, а також, після прочитання детального пояснення в методичних вказівках, був реалізований тест Міллера-Рабіна на простоту.

2. Реалізував генерацію двох пар чисел p і q, такі щоб pq < p1q1. Тут використовувалася функція генерації чисел, що містить в собі перевірку на простоту за допомогою теста Міллера-Рабіна.

3. Далі виконується функція генерації ключових пар для RSA, ка повертає секретний ключ (d, p, q) та відкритий ключ (n, e) для двох абонентів. Приклад генерації цих ключів наведений нижче:

Ключі для абонента A:

d:
18270776606355864965647944226704900037900003502831212824803668620205334211066
56114492337588529243971389914305657252679014534830374431076809440673765654275

p:
76106562456698229525961159911912710307442903556507935000566163289172996455737

q:
84223676252732454804419821433992112496172513806580506519622580730889003040263

n:
64099744770613140697754259879287031681350033066770362252550258977622725655030
12918158961032440884485891830452403203988955201056469563248028121916508338831

e:
16269038615551313436631125836909662665383100526095648699998165809417015399676
31439403873228465274550308144420259261974325028868449073538056314480586409019

Ключі для абонента B:

d:
77274382101952292153975408469479705322982883659330933363584823522065455151598
80932927213599799928958827716579101810508639003657098589219946598371702821401

p:
76429885701504443057760041840253242601528959553178181444355237487198246564339

q:
11243778746319764364002219062942277620947141607251458467439366949320494145349
1

n:
85936072443422451086951347361609171418481123117016005969187795840401014557710
68130947314307860877125386848402051660296005284220163923225166289231707657449

e:
113288573623136373848787128139494188468186624506966983300497014009314511420129454578669100623133542388800533435715840611712907353023445933814944220469988 1

Труднощі, що виникли:

Не вдалося реалізувати програми шифрування, розшифрування і створення повідомлення за допомогою цифрового підпису (пункт 4), а також протокол розсилання ключів (пункт 5).

Висновки

Отже, я ознайомився з різними методами тестування чисел на простоту, був реалізований тест Міллера-Рабіна. Ознайомився та практично реалізував генерацію ключових пар асиметричної криптосистеми RSA. Отримав теоретичні знання про те, як працює криптосхема RSA.