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

Виконали:

Студент групи ФБ-84

Мороз Дмитро

Студент групи ФБ-84

Яловчук Михайло

Перевірив:

Чорний О.М.

Київ - 2020

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.

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 1 1 p , q довжини щонайменше 256 біт. При цьому пари чисел беруться так, щоб $pq \leq p1q1$; 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.

p=8f90d50991379ab69cb473691bb516359eca267ce5fe81fe8e7bd22ae575b70f

q=d2015b7aff9416efd424f888800942e4105569199ed5ae5ac6263df87f4c8d29

n=75c591a0243d6c1c026e01f0edc303ed614ce611abbd28d75438eac1991ecaf681f217a22d4649deaa499c329a5dce88ca84a4a40bc63ad4ce09b78b5f219467

e=6542c5f429a41e94ca28378597edb4331fa5a25234b474557f7d234e13ade1aba04467bccf3297b92be5a203c0236b4d16509987f511b93188744cff2324e8f3

d=570c53d5d0d9628dc695e81baa4b1d208a6799649f33dcb5a97393f6e8ffbe5536ac09ca731671d64040f324eac0a4f5e44ec1c6783da76cad91e302f805a55b

p1=9fc5d890b0c17be5a2756d41fda35fdbcccdf96380b12ad306ab3297bbeb31db

q1=f4d07ea4d8c1d2a82f3a40a74e0607d8f3fed87804cf5ba67b1ffdc453d4b615

n1=98cab23420a5e1579d8950e7ec4e919426817154cc545736d9ba69692f72316e806ffaf97884364000bf9aea01aad7078e452240e9f2318eec7480a6ea18c8f7

e1=3ecc848f0bb7223d38acfdee18bff3497c910c9ff0b97031492da5a457d44f351af84d5454996ea1ef587d2cb3907041ad8bdaa687f16dd172e79f6276f2be51

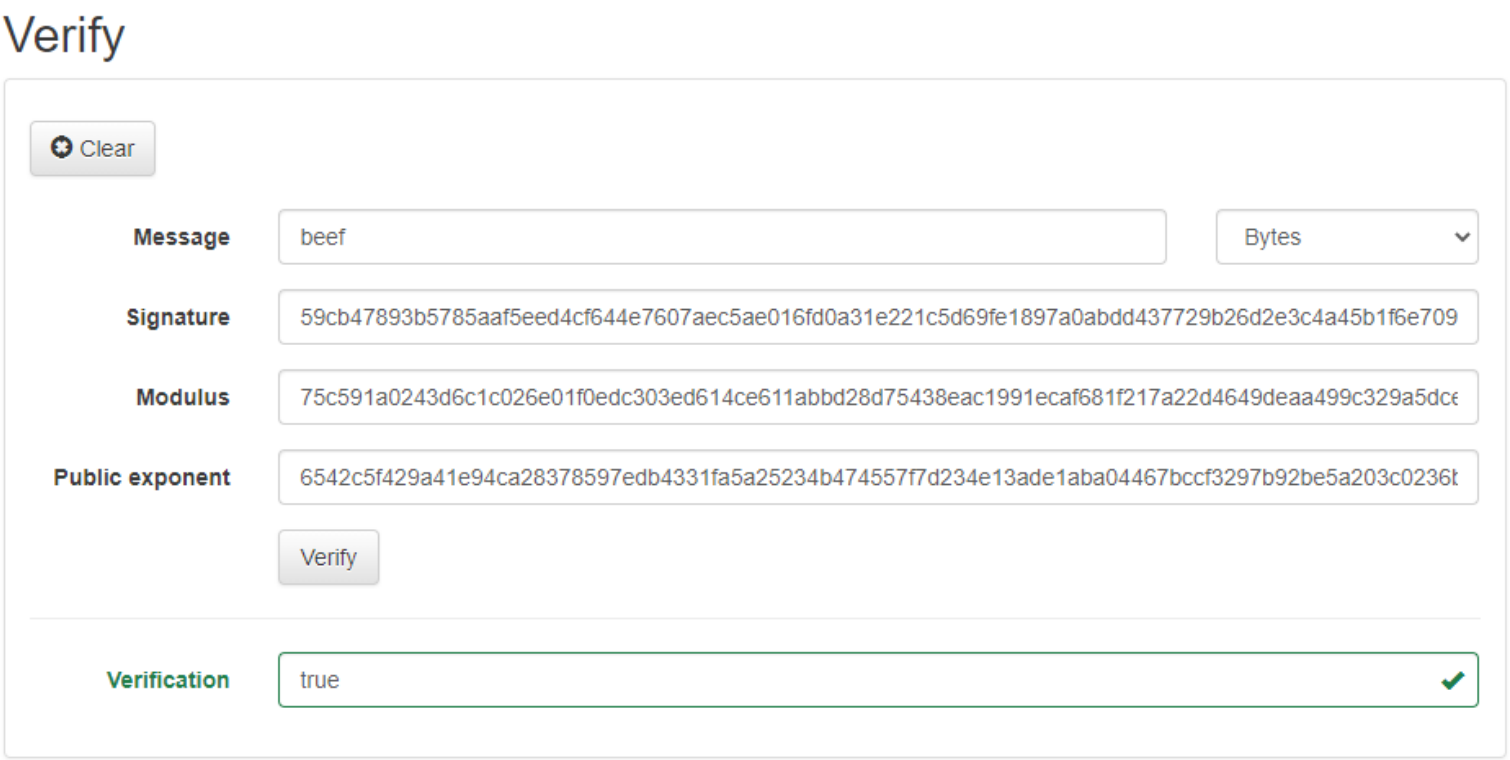
d1=42162ca02a8aa0f0274f0acdc4f522f47bb62d471a764563e3a5f8cf2574eccb6f7188c163e7229a5543039951d824fceba75a9a6f08eeb357bf485751cbbb69

Чисельні значення прикладів ВТ, ШТ, цифрового підпису:

ВТ: **beef**

ШТ: **56cfd4c6747295d7a2270d71c6e7c299ba7f217520253b682f944b1b8f99cfaa9e30e6ff45a6bb324ae288ff4dc93af8f8f6a77eb4b4e3312b36610985db5427**

Цифровий підпис:59cb47893b5785aaf5eed4cf644e7607aec5ae016fd0a31e221c5d69fe1897a0abdd437729b26d2e3c4a45b1f6e709bcf03674b3ce7711074c5d6db56b49b3d1



Висновок

В ході лабораторної роботи ми ознайомились з тестом перевірки числа на простоту (функція Міллера- Рабіна), і методами генерації ключів для асиметричної криптосистеми типу RSA. Ознайомились з системою захисту інформації на основі криптосхеми RSA, організація з використанням цієї системи засекреченого зв'язку й електронного підпису, вивчення протоколу розсилання ключів.