Криптографія

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Вивчення криптосистеми rsa та алгоритму електронного підпису. ознайомлення з методами генерації параметрів для асиметричних криптосистем

Виконали: Мельниченко Богдан, Захаренко Нікіта

Варіант: 8

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 p1 , q1 довжини щонайменше 256 біт. при цьому пари чисел беруться так, щоб $pq < p1q1$. p і q – прості числа для побудови ключів абонента a, p1 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), (e1, n1) 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 $0 < k < n$.

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.

Знаходимо випадкові прості числа заданої довжини

```
on310/python.exe "d:/5_sem/crypto/4/main (2).py"
Find prime
215331779893698042654168307789723145557929573188939849309868147245758616
952689
Find prime
200256762833485432043487590539043011510169119376707608052736178285694435
452343
Find prime
209287205622133938507456719638915446655567716915992969215601996365463601
961679
Find prime
139092670562160407244121027781049308868915097766756884825045858179184366
784581
Find prime
207212023966803707268548743491937520804961183699024267961042098708284822
685763
Find prime
160797056651206082381033644329780608610452747290017176473966014756650059
```

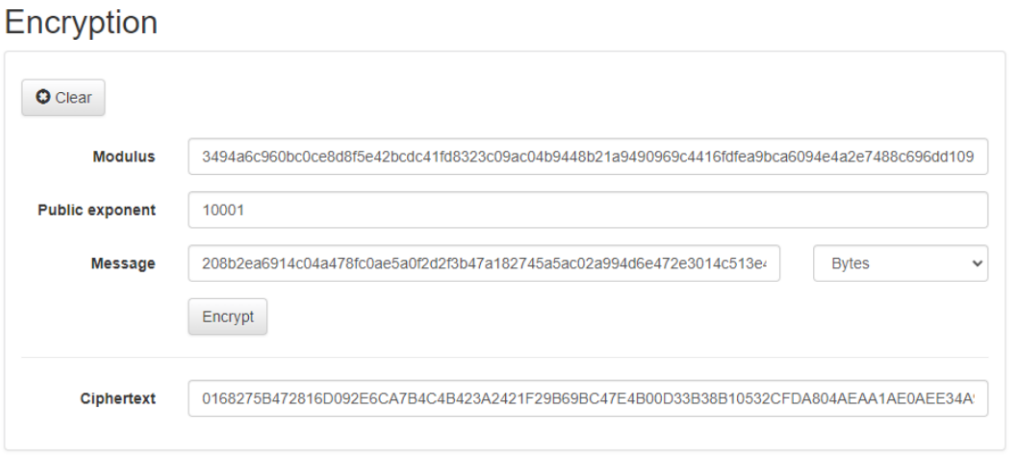
2-3. За допомогою цієї функції згенерувати дві пари простих чисел p, q і p1 , q1 довжини щонайменше 256 біт. при цьому пари чисел беруться так, щоб pq < p1q1. p і q – прості числа для побудови ключів абонента a, p1 і q1 – абонента b.

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), (e1, n1) та секретні d і d1.

```
Sender public and private keys: 299744657072061542318231904427425366648551
929971193101326328154175981679383171155467294577640444234922670312452877
737815557378653494434157957909389834574823 (0x23c5022ecba14f2dd5a3b400884
535cf72d9039cbefe3f0e95145a36570df604206654378a58fd43f60c464af2ec61b7d0738
788eeaf9b4b5ba02e821d5addbe7)
65537 (0x10001)
Receiver public and private keys: 360250055108867220540222231133392849344
124489617914689277183765023978513998108150717465559803769119325933014046
112983709614104473280757303384448861628323 79 (0x2afd68f15f4b5ab1a5b716efc
63209ba3d1416b2872bc157a3c1cfb0b3700f96ff329dadc2d75e2313b9ad5e1847e78dd39
797a40df2f1b44647ad1bc1bd07bfb)
65537 (0x10001)
Original message: 22426333415914174191948149753579782862699048685675369798
543231679172796991970418379718951816150600643029326682094090017373454966
373467968102004918464021371 (0x1ac31a26307d9a4a1f4de3098b92a78fcae638b9412
d785b09d9a438847c14317290523cbfbda78b666dca2ebf839cd3923b337fe2d0ccb03c4e5
3c536764037b)
*  ‾  ^  ‾  *
```

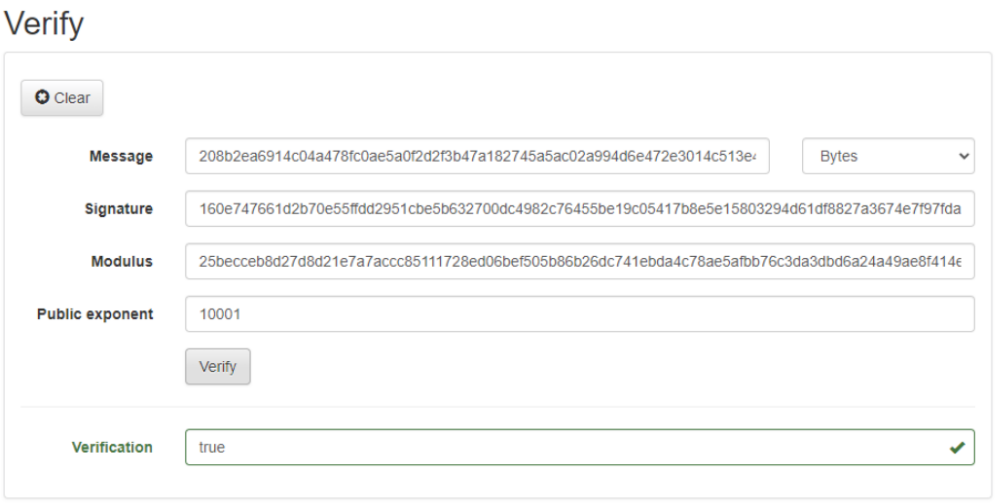
4.Написати програму шифрування, розшифрування і створення повідомлення з цифровим підписом для абонентів a і b. кожна з операцій (шифрування розшифрування, створення цифрового підпису, перевірка цифрового підпису) повинна бути реалізована окремою процедурою, на вхід до якої повинні подаватись лише ті 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.

На першому слайді ми беремо хешований наш текст публічна експонента за замовчуванням(10001в 16-ричній) або (65537 у десятковій)(просте число фермі) модуль це n нашого отримувача на виході отримуємо шифротекст порівнюємо його з тим що дає наша програма.



Співпадає значить все добре.

Тепер перейдемо до другого слайду, спочатку ми вводимо наше повідомлення, потім сигнатуру а потім модуль надсилаючого, після чого дивимось чи пройшли верифікацію.



Висновок: отже, під час виконання цієї лабораторної роботи, ми ознайомились із різними методами перевірки простоти чисел та з методами створення ключів для асиметричної криптосистеми rsa. ми практично вивчили систему захисту інформації, яка ґрунтується на криптосхемі rsa. ми створили засекречений зв'язок та електронний підпис за допомогою цієї системи, а також вивчили протокол розподілу ключів. ми також реалізували протокол конфіденційного розподілу ключів за допомогою мови програмування python3 на відкритих каналах зв'язку, з підтвердженням автентичності відправника