МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ "КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ"
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Лабораторна робота №4
З Криптографія

Виконали студенти группи ФБ-81
Дубравська О.
Зозуля А.

Перевірив
Чорний О.

Київ – 2020
29.11

Мета

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.

Завдання

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 $pq \le p_1 q_1$; p і q – прості числа для побудови ключів абонента A, 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 A 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 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 $0 < k < n$.

Хід роботи

Alice:

p=90125341376161157443307814610142514142915127340968635928831128279428461467219

q=71905886345894501762309113489002826242575714591413236478187282135622906526727

Bob:

p=79985531644043628695735211079564786126514402258119300312146881810596334593747

q=91575717259749741233226361863451629500648203012813346350299723092222412376307

Программа не зберігає відхилені значення кандидатів що не пройшли перевірку простоти тому я просто нагенерив рандомних чисел:

655543931817531467264538774048170787926449922579219298106717679228652638224615979950575589627076523322955350804257754166155923530671170731530545697133745397267025093326079091953772255218539346103861094659084370832555962548388447657703808242181953569806251114615241850713444309759024360340985461257055443368197993744610702050081991533829438906463473817501145320577905485160104928047085788522071480826173570762232934368649834189775846726477027502915557255385911691100116489933943073237226598882265923765458784352902897479916690641652658626067115335906101715964155278392257457768719671416120515816831035726149419929836721822010504642316048041640269063430663812009264149939751104947783331463219230731088671426753594181721409728576403172996803051031019462775624646194243441434489604239702279159139612439882119683461228545537064925565083703125696416184222679870118770975867049529200441877457677646643491003783309098052319637436774426198263987986365362099956156093007103986836717028636031045922592054996995259497699843928619277178326218829382801458452866869678325405049548182668136411122719649341172441767205164629056572523206549413404383489722639368516819615518940759024822557245255301095814871602949835614175049733470432372080488079752080409795498054694449590064528272509777506470624192783090266217899323328719205859336957135493432458159872949569143896057608781610220676231753038956892120464526377181262038257347254236790815359608008737908337063603166211494898941077029011948128460788271169697087001675897827472631124478822959605433965541187613217276379650179705513359865156425792133370179058027123326843517223068542883219975197504664130468112657111203568110309289471194280131184824467408909917423829827
Приклад:

Исходное сообщение:
964934117244176720516462905657252320654941340438348972263936851681961551894079

ШТ Алиса -> Боб:
421158611325380082344080885520275038888648669521452944278173658493177193316674904431187662274400354880895718292177198643191305959848584883553506788572741 1

ШТ Боб -> Алиса:
464551340229495998295665032537307933119391308816733937865048304156206026627847125866655863335726608106182248711869922071768807365917877098651438103741937 5

Подпись Алисы:
381736941606149569093531758500880445929447212004481065941161527799567274227122824294030891883242278525035586026070746147793180052501404247291972565921442 2

Подпись Боба:
346577797675524909335950627116577383207602981242249698529377143990635219926785699945497680172364581679079494212973762177580621440536800432814705364860185 6

Кроки конфіденційного розсилання ключів із підтвердженням справжності:

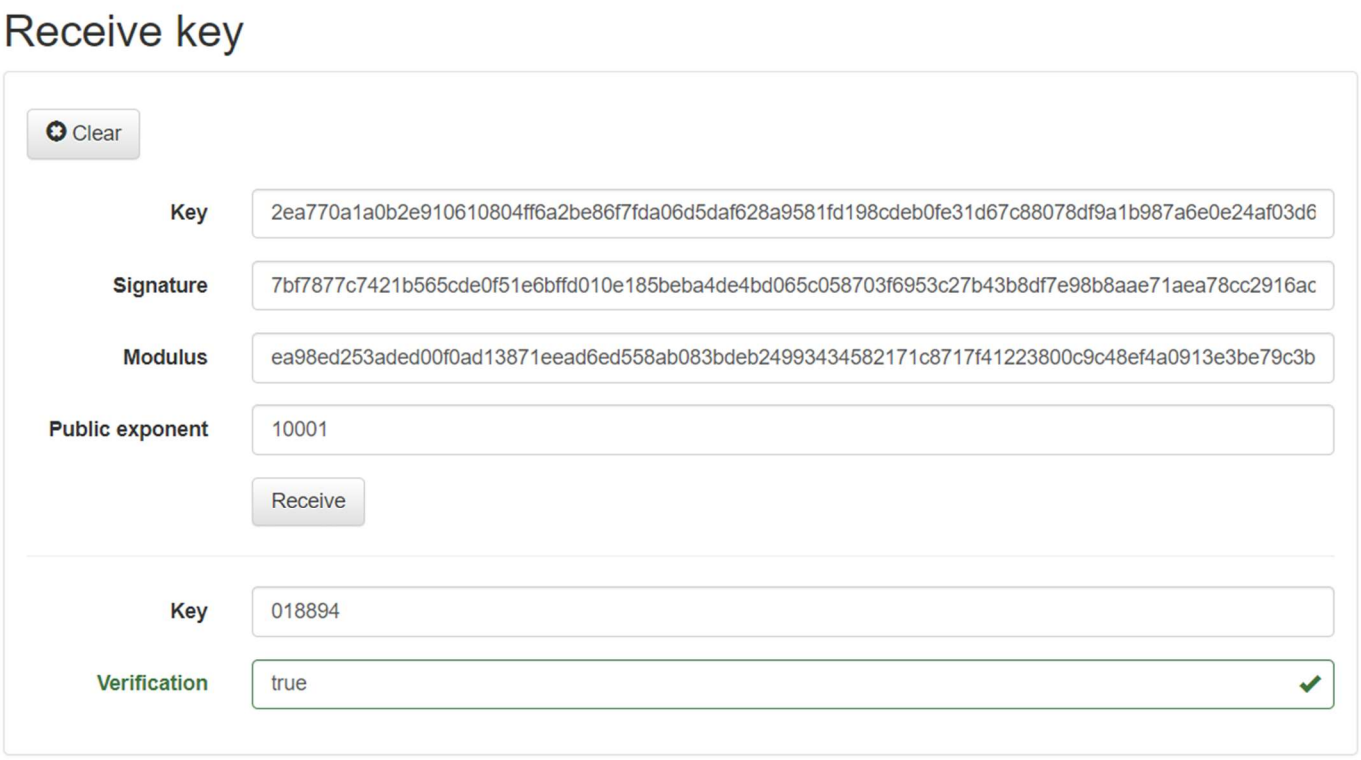
1. Аліса обирає повідомлення k та за допомогою функції send() формує пару k1 та s1
    a. Ширування k->k1 за допомогою e, n Боба
    b. Підпис k->s за допомогою d, n Аліси
    c. Ширування s->s1 за допомогою e, n Боба
2. Боб дешифрує k1 та s1 та підтвердження відправника за допомогою receive()
    a. Дешифрування k1->k за допомогою d, n Боба
    b. Дешифрування s1->s за допомогою d, n Боба
    c. Перевірка відправника s->k за допомогою e, n Аліси

Приклад виконання протоколу обміну ключами із сайтом:

```
k=100500
arr = Computer.send(k, Site)
k1 = hex(arr[0]).split('x')[-1]
s1 = hex(arr[1]).split('x')[-1]
e = hex(Computer.e).split('x')[-1]
n = hex(Computer.n).split('x')[-1]

URL = 'http://asymcryptwebservice.appspot.com/rsa/receiveKey?key={}&signature={}&modulus={}&publicExponent={}'.format(k1, s1, n, e)
print(URL)
```

Та вручну:



Висновки

- В результаті роботи отримано навички по роботі з асиметричною криптографією, RSA.
- Завжди буде швидше (як по часу написання так і по часу виконання) використати бібліотечну функцію аніж намагатися робити свою
- При роботі із великими числами слід спочатку ознайомитися з особливостями мови програмування для них (адже різниця порівнянно із звичайними числами може бути критичною)